55901

ISTANBUL TECHNICAL UNIVERSITY ★SCIENCE AND TECHNOLOGY INSTITUTE

MOBILE ROBOTS

M.Sc. Thesis

Mech. Eng. Bilin AKSUN

Date of Submission	:	**January 15, 1996**
Date of Defense	:	**January 31, 1996**
Advisor	:	**Assoc. Prof. Dr. N. Aydın HIZAL** [illegible]
Other Jury Members	:	**Prof. Dr. A. Talha DİNİBÜTÜN** Talha
		Prof. Dr. Ahmet KUZUCU [illegible]

JANUARY 1996

FOREWORD

Mobile robot concept is a very new field of robotics and the state-of-the-art in the area of mobile robots reflects the process of evolving the idea of autonomy, as well as the level of technology. It would be desirable to have an automated device which can move and perform useful operations with no human involvement over substantial time intervals.

In this study, the main object is to give an idea of what mobile robot is, which locomotion systems, navigation systems, motion planning strategies and control structures are used for their guidance. In addition, a simulation of a point-to-point (PTP) motion control of a three-wheeled differentially steered mobile robot has been studied.

I would like to thank Associated Professor Aydın Hızal and Professor Ahmet Kuzucu for their suggestions, considerable support and consultants in my thesis.

Bilin AKSUN

January, 1996

TABLE OF CONTENTS

LIST OF SYMBOLS

M	: Mass of the vehicle.
I_v	: Yaw moment of inertia of the vehicle
L	: Linear distance between front wheels.
V	: Actual linear velocity of the vehicle.
V_{ref}	: Reference linear velocity
θ	: Direction angle (orientation) of the vehicle.
θ_{ref}	: Reference direction angle
x	: Actual position in x-direction
y	: Actual position in y-direction
x_{t_i}	: Target point in x-direction
y_{t_i}	: Target point in y-direction
T_m	: Torque developed by motor
I_m	: Moment of inertia of the motors of the mobile robot
b_m	: Viscous friction coefficient of the motors of the mobile robot
n	: Reduction ratio
ω_w	: Angular velocity of the wheel
ω_m	: Angular velocity of the motor
r	: Radius of the wheels of the mobile robot
I_w	: Moment of inertia of the wheels of the mobile robot
B_w	: Viscous friction coefficient of the wheels of the mobile robot

F	: Traction force of the wheel
ΔT	: Unit time
K_{p_v}	: Proportional gain for linear velocity controller
K_{i_v}	: Integral gain for linear velocity controller
K_{p_θ}	: Proportional gain for direction angle controller
K_{v_θ}	: Derivative gain for direction angle controller
u_1	: PI controller output
u_2	: PD controller output
u_R, u_L	: System inputs

SUMMARY

In this thesis, the main object is to give an idea of mobile robot concept because mobile robotics is a very new field of robotics research in our country and also technologically developed countries.

In chapter 1, definitions of mobile robot, design considerations, a brief overview of mobile robot projects, and the application areas where mobile robots are used have been explained in details.

In chapter 2, the basic locomotion systems of mobile robots have been concerned. According to this consideration, mobile robots can be classified as:

Wheeled mobile robots,
Tracked mobile robots
Legged mobile robots
Articulated body structured mobile robots
Combinations of basic configurations

The drawbacks and advantages of each locomotion system have been studied.

In chapter 3, basic navigation and multisensory navigation systems used for mobile robots so far have been studied by giving the real application examples.

In chapter 4, global path planning, local path planning and collision avoidance have been concerned including commonly used path planning techniques for mobile robots.

In chapter 5, control structures for autonomous mobile robots have been studied. These control structures are divided into mainly two categories:

Functional Decomposition
Behavioral Decomposition

Under this classification, significant control structures belong to these groups have been explained in details.

In chapter 6, simulation of a point-to-point (PTP) motion control of a three-wheeled differentially steered mobile robot has been studied.

ÖZET

MOBİL ROBOTLAR

Bu tez çalışmasındaki ana amaç, günümüz yüksek teknolojisinin dünyada ve ülkemizde oldukça yeni bir araştırma konusu olan mobil robotlar hakkında bir fikir vermektir. Bu amaç doğrultusunda da mobil robotların tarihsel gelişimleri, belli başlı projeler, uygulama alanları, temel hareket sistemleri, gezinim sistemleri, yörünge planlama ve planlama teknikleri ve kontrol yapıları genel bir çerçevede incelenmiştir. Ayrıca dar bir kapsamda, üç tekerlekli diferansiyel tahrikli bir mobil robotun noktadan noktaya hareket kontrolünün bilgisayarda simülasyonu gerçekleştirilmiştir.

Mobil robot, en genel biçimde, verilen bir yön ve konum boyunca hareket edebilme yeteneği olarak tanımlanabilen mobilitesine göre otomatik olarak hareket edebilen serbest programlanabilir veya özerk bir araçtır. Diğer bir deyişle, mobil robot, belirli bir serbestlik derecesine sahip olarak tamamiyle kontrol edilebilen bir şekilde hareket edebilen makinalar olarak ifade edilmektedir. Endüstriyel robotların çoğu, sabit bir tabana monte edilir ve hareket edemezler. Çalışma hacimleri robot eklemlerinin ulaşabileceği hacimle sınırlıdır. Bu nedenle de bu tip robotlarda yapacakları işin kendilerine getirilmesi gerekmektedir. Mobil bir robot ise, belirli bir görevi yerine getirebilmek amacıyla işin bulunduğu kısma gidebilmektedir.

Mobilite konusunda endüstriyel uygulamalardaki ilk adım 1975 yılında Olivetti tarafından sunulan kızaklı robot tipidir. Bundan sonraki gelişme, fabrikaların büyük bir bölümüne ulaşabilen tekerlekli araçlardır. Uzaktan kumandalı olarak bir operatör tarafından dolaylı biçimde yönlendirilen ve hareketi kontrol edilen robotlardır. Teleoperasyon işlemi ile çalışan bu robotlar tehlikeli bölgelerde çok iyi bir performans sağlamışlardır ancak operatöre ihtiyaç duymaları nedeniyle pahalıdırlar. Teleoperasyonla çalışan robotlardan sonra Kendiliğinden Yönlendirilmiş Araçlar veya İngilizce karşılığının baş harflerinin alınmasıyla adlandırılan AGV'lere geçiş tabii bir adımdı. Kendiliğinden Yönlendirilmiş Araçlar sürücüsüz forklifte benzemektedir. Fabrikanın bir köşesinden diğer bir köşesine yükleri götürebilmekte ve genellikle belirli (sabit) bir yol izlemekle sınırlandırılırlar. İzleyeceği yörünge, aracın izlemek üzere programlandığı yere gömülü bir tel veya yere çekilen boyalı bir hattır. Kendiliğinden Yönlendirilmiş Araçlar, gömülü teli izlemek üzere radyo frekanslı sensör veya görünen boyalı hattı izlemek üzere optik algılayıcılar kulllanırlar.

Kendiliğinden Yönlendirilmiş Araçlardan sonraki en arzulanan aşama tamamiyle özerk mobil robotlardır. Özerk mobil robotlar, robotik sistemlerdeki gelişmenin bir sonucu olarak karşımızdadır. Bu aşamada, belirli bir görevi insan müdahalesi gerektirmeden yerine getirmek amacıyla tasarımlanmış tekerlekli, paletli

veya bacaklı hareket sistemlerine sahip özerk robotlar yer almaktadır. Özerklikten kasıt, robotun değişken durumlara karşı kendi kendine karar verebilmesi ve bu kararları uygulayabilme yeteneğidir. Özerk bir mobil robotun, dinamik değişkenlik gösteren bir ortamda gerekli kararları verip uygulayabilmesi için algılayıcıları vasıtasıyla çevresinden mümkün olduğu kadar bilgi toplaması gerekmektedir. Özerk bir mobil robotun karşılaması gereken özellikler aşağıda belirtilmektedir :

- Robot, kapalı bir şekilde ifade edilmiş bir görevi, giriş olarak alarak net hareket planlarını formüle edebilmeli ve meydana getirebilmelidir.
- Robot, meydana getirdiği hareket planlarını bağımsız bir şekilde yerine getirebilmelidir.
- Çevresini algılayabilmeli ve mevcut amacıyla çevresi arasında tutarlılık sağlayabilmelidir.
- Çevresiyle sürekli iletişim ve bağlantı halinde olmalıdır.
- Beklenmedik olaylara ve durumlara karşı tepki verebilmelidir.
- Yetenek geliştirebilmek için aktif veya pasif öğrenme kabiliyetine sahip olmalıdır.

Mobil robotlar konusundaki ilk çalışmalar 1970'lerin başlarına rastlamaktadır. Bu konudaki ilk çalışma Stanford Üniversitesinde gerçekleştirilen SHAKEY mobil robotodur. İlk çalışmalarda planlama ve problem çözme yetenekleri üzerinde durulmuş robotun çevresini algılaması, algılayıcılar ve kontrol üzerinde fazla durulmadığı için istenen sonuçların alınamaması mobil robotlar üzetinde yapılan çalışmaları yavaşlatmıştır. 1980'lerden itibaren endüstriyel robotlardaki, bilgisayarlardaki ve görme sistemlerindeki gelişmeler neticesinde çalışmalar yeniden canlanarak günümüzde yüksek teknolojinin en önemli araştırma alanlarından biri olarak yer almaktadır. Mobil robotlar üzerinde günümüze kadar gerçekleştirilmiş ve hala devam etmekte olan çalışmalara Bölüm 1'de ayrıntılı bir şekilde yer verilmektedir.

Mobil robotlar;

- Fabrika otomasyon projelerinde, sabit tabanlı robotlar veya makinalar arasında malzemeleri ulaştırma amaçlı,
- Yangın, deprem veya nükleer santraller gibi tehlikeli ortamlarda kurtarma, tetkik, bakım v.b görevleri yerine getirme amaçlı,
- Uzay ve gezegen keşiflerinde geniş bir alanda çevre hakkında veri toplama amaçlı.
- Sualtı araştırma ve çalışmalarında bakım ve tetkik amaçlı,
- Sağlık alanında, hastanelerde hastalara ilaç, yemek ve mektupların dağıtımı gibi görevlerde veya özürlü insanlara yardım amaçlı,
- Geleceğin meydan savaşlarında askeri alanda mayın temizleme, döşeme, cephane ve diğer gereçlerin savaş alanı birimlerine iletimi v.b. amaçlı,
- Hapishanelerde ve fabrikalarda güvenlik amaçlı,
- Evlerde temizlik, yemek servisi ve günlük işlere yardım amaçlı,

olarak kullanılabilirler.

Mobil robotun hareket sistemi robotun mobilitesine imkan veren alt sistemidir. Mobil robotlar hareket sistemlerine göre:

1)Tekerlekli

2) Paletli

3) Bacaklı

4) Eklemsi yapılı

5) Temel hareket sistemlerinin kombinasyonu

şeklinde gruplandırılabilirler.

Teoride seçim güç ihtiyacına, maliyete, istenen hareket karakteristiklerine ve robotun hareket etmesi gereken yüzeylere dayanmaktadır, ancak pratikte seçim genellikle tekerleklerdir. Tekerlekler, düz veya sınırlı bir eğime sahip az pürüzlü yüzeylerde iyi performans ve çekme kabiliyetine sahiptirler. Bu sebeple de en fazla kapalı alanlarda tercih edilmektedirler. Aksi takdirde özellikle engebeli arazilerde ve açık alanlarde paletler tercih edilmektedir. Bacaklı robotlar, birçok arazi tipinde çalışabilecek ve merdiven tırmanabilecek yeteneğe sahiptirler. Fakat tasarımları, bacak kontrolleri, denge için gereken hesaplamalar oldukça uzun ve karmaşıktır. Bacaklı mobil robotların çektikleri güç o kadar büyüktür ki deneysel modeller üniversitelerde veya AR-GE Laboratuvarlarında görülmektedir. Eklemsi yapılı mobil robotlar ise düzgün olmayan arazi ve dar çalışma alanlarında uzun ve eklemli gövdelerini yerin topografisine adapte edebilmektedirler. Ayrılabilir bir konstrüktif yapıya sahip olmaları nedeniyle de arıza halinde olan parçaları sökülerek yerlerine sağlam parçalar monte edilebilmektedir. Temel hareket sistemleri olan tekerlek, palet ve bacak mekanizmalarının yetersiz kaldığı koşullarda ise tekerlek-bacak, tekerlek-palet gibi kombine hareket sistemlerinden faydalanılabilmektedir.

Mobil robotlarda gezinim, aracın hareketinin bir noktadan diğer bir noktaya başarılı bir şekilde yönlendirme işlemi olarak tarif edilmektedir. Gezinim sistemi mobil robot için en temel bir yapıdır, çünkü hareketi sırasında sürekli olarak çalıştığı alanda ve etrafında yer alan çeşitli nesnelere göre nerede bulunduğunu bilmesi, gideceği hedefe bir aksilik olmadan ulaşabilmesi için gerekmektedir. Bölüm 3' de mobil robotlarda kullanılan çeşitli gezinim sistemleri hakkında bilgi verilmektedir.

1) Kestirme Hesap Metodu:

Oldukça ucuz bir gezinim yöntemi olan Kestirme Hesap metodunda, robot tarafından alınan yolun yönü ve mesafesi ölçülmekte ve bu ölçümler başlangıç konumuna eklenmektedir. Katedilen mesafenin ölçümü için tekerleklerin dönüş sayılarının sayılmasıyla (odometre) beraber yön bulunumu için jiroskop kullanımı veya diferansiyel olarak tekerlek döndürülmesi en sık kullanılan yöntemdir. Ölçüm hatalarından, tekerleklerdeki kayma ve tekerlek elastisitesi nedeniyle hareket sırasında giderek artan hatalar meydan geleceğinden, hatayı kabul edilebilir bir küçüklükte tutmak için belirli periyodlarda gezinim sisteminin kalibrasyonu gerekmektedir.

2) Inertial (Ataletsel) Gezinim :

Inertial gezinim sistemleri, havacılık gezinim sistemlerinden adapte edilmiş bir gezinim sistemidir. Bir çeşit kestirme hesap metodu olup ölçülen ivmenin iki defa integrali alınarak iki veya üç boyutlu olarak konum hesaplanmaktadır. İvme ölçümü için akselerometre kullanılırken, referans takımına göre ivmenin yönünü bulmak için jiroskopdan faydalanılmaktadır. Inertial gezinim sistemlerinde akselerometre ve jiroskop bir takım halinde monte edilmişlerdir. Oldukça pahalı donanımlardır.

3) Görme ve Doğrudan Görüntüleme Sistemleri:

Görme ve doğrudan görüntüleme sistemleri özellikle mobil robot gezinim uygulamaları için çok uygun sistemlerdir. Robotun etrafında bulunan nesnelere uzaklık bilgisi önemli bir veridir çünkü bu bilgiden faydalanılarak mobil robotun konumu, mevcut serbest alanlar ve engellerin mevcudiyeti tayin edilebilir. Görme algılayıcılarından faydalanılarak Stereo Uzaklık Ölçme yöntemi en çok kullanılan yöntemlerden biridir. Bu yöntemde, nesneye olan uzaklık elde edilen iki görüntü vasıtasıyla saptanmaktadır. İki görüntü, iki farklı kameradan alınabileceği gibi tek bir kameranın hareket ettirilmesi ile de sağlanabilir. Uzaklık ölçümü için doğrudan görüntüleme sistemlerinden de faydalanılmaktadır. Optik ve lazer kullanılarak oluşturulan uzaklık ölçme sistemleri hakkında incelemeye tez kapsamında yer verilmektedir. Mobil robotlar için görme sistemlerinden faydalanılarak yol-izleme algoritmaları da geliştirilmiştir. Bu sayede robot üzerinde bulunduğu yol ve ona göre konumu hakkında sürekli olarak bilgi alarak gezinimini sürdürmektedir. Bu algoritmalara ait açık-saha uygulamalarına değinilmiştir. Hareket kontrolünde kullanılmak üzere tasarımlanmış görme sistemlerinin geliştirilmiş bir yapısı olan Dinamik Görme yapısı da ele alınmıştr. Günümüz teknolojisinde görme ve doğrudan görüntüleme sistemleri büyük bir hızla gelişmekte ve bunun sonucunda da fiyatlarında düşme kaydedilmektedir, ancak hala karmaşık ve zaman alıcı hesaplamalar gerektiğinden mobil robotun hızını sınırlayıcı bir etkiye sahiptir.

4) Ultrasonik Sistemler:

Ultasonik sistemlerden faydalanılarak da mobil robot gezinimi için gerekli uzaklık bilgisi elde edilmektedir. Bunun yanında ultrasonik sistemler mobil robotun çevresinin haritasını oluşturma işlemlerinde de kullanılırlar. Algılayıcının küçük adımlarla hareket ettirilerek mobil robotun çevresindeki nesnelerin konumlarını gösteren haritalar elde edilebilir. Gezinim sırasında, robot bulunduğu ortam hakkında hiçbir ön bilgiye sahip değilken ultrasonik algılayıcılar vasıtasıyla robotun bulunduğu ortamın sonar haritası elde edilebilmektedir. Ultrasonik sistemler, görme sistemlerine nazaran daha ucuz olmakla beraber zayıf yönlenme, sık hatalı okuma ve geniş açılı yansımalar gibi birtakım dezavantajları mevcuttur.

5) Nirengi Metodu:

Yön bilgilerinden faydalanılaraktan iki boyutlu konum belirleme işlemi Nirengi metodu olarak tanımlanmaktadır. Yön bilgisi belirli istasyonlara yerleştirilen işaret alıcıları vasıtasıyla saptanmaktadır. İşaret alıcıları olarak infrared, ultrasonik, lazer ve hatta çubuk-kodlayıcılardan dahi faydalanılabilmektedir. Belirli stratejik noktalara yerleştirilen alıcılarla oldukça doğru yön ve konum bilgileri elde edilmekle beraber robotun yol alacağı tüm alanda gerekli yerlere yerleştirilmeleri zorunluluğu maliyeti arttıran bir etkendir.

6) Çok-Algılayıcılı Gezinim Sistemleri:

Çok-algılayıcılı gezinim sistemleri robotun çevresi ve çevresine göre konumu hakkında daha güvenilir bilgi almasını olanak veren gezinim sistemleridir. En basit, çok-algılayıcılı gezinim sistemi olarak Kestirme Hesap metodu ile Optik İşaretlerin Saptanması yönteminin birlikte kullanılması verilebilir. Odometre tek başına giderek artan hataların sonucunda hatalı ölçümler verirken Optik İşaretlerin Saptanması yönteminden faydalanılarak aracın yol boyunca belirli aralıklarla yerleştirilen optik işaretler vasıtasıyla aracın konum kalibrasyonu temin edilebilmektedir. Çoklu gezinim sistemlerine sahip mobil robotlardan Hilare ve Stanford Mobi robotları örnek olarak verilebilir. Hilare robotunda ultrasonik, görme istemeleri, infrared nirengi ve odometre yöntemleri çok-algılayıcılı gezinim sistemini oluşturmak üzere kullanılmıştır. Stanford Mobi robotunda ise ultrasonik, görme sistemleri, dokunsal algılayıcılar ve odometre yönteminden faydalanılmaktadır.

Özerk mobil robotlarda hareket planlaması problemi, en genel biçimde aşağıdaki şekilde belirtilmektedir.

1. Mobil robotların başlangıç durumlarının verilmesi.

2. Mobil robotların arzulanan son durumlarını verilmesi.

3. Robot hareketi üzerinde kısıtlamaların verilmesi ve robotun bu kısıtlamaları da gözönünde bulundurarak başlangıç noktasından istenen son noktaya engellerin bulunmadığı bir yol bulmasıdır.

Hareket planlaması problemi kısaca Yörünge Planlama olarak adlandırılabilir. Yörünge planlama global ve yerel olmak üzere iki bölüme ayrılmaktadır.

Global Yörünge Planlama:

Global yörünge planlama, gerçek ortamın önceden öğrenilmiş olarak basitleştirilmiş halde tanımlanmasını gerektirmektedir, bu nedenle de hareket anında oluşan değişiklikleri yansıtmamaktadır. Kısaca, global yörünge planlama robotun nerede ve hedefinin neresi oduğuna karar vermektedir. Bu karar genellikle robotun hafizasında yer alan çalışma hacminin genel bir haritasına göre alınmaktadır. Robotun bulunduğu andaki konumu, haritada yenilendiği anda mobil robot bulunduğu yerle hedefi arasında bir yörünge oluşturabilmektedir. Eğer robotun etrafında hiçbir engel yoksa yörünge doğrudan oluşturulabilir, engellerin mevcut olması halinde yörünge planlama algoritmalarından uygun bir tanesi kullanılarak hedefe giden en kısa yol saptanabilir. Bulunan yörünge onaylandığı anda robot hedefe doğru hareket edebilir.

Yerel Yörünge Planlama:

Yerel yörünge planlamada ana amaç robota beklenmedik engellerin etrafından en iyi yolu bulmaktır. Diğer bir deyişle beklenmedik olaylara reaksiyon olarak gerçekleştirilen taktiksel bir yaklaşımdır. Global yörünge daha önceden bilimesi gereken bir ortama ihtiyaç duyarken, yerel planlamada robotun etrafında meydana gelen değişikliklerin planlama meydana getirilirken bilinmesi gerekmektedir. Yerel yörünge planlamada daha doğru sonuçlar elde edilmektedir. Bunun başlıca sebebi de mevcut algılayıcıların daha küçük bir bölge üzerinde çalışmalarıdır. Yerel yörünge planı ile elde edilen yörünge genellikle kısa olduğundan ilk bulunan yörünge kullanılabilmekte ve ekstra bir optimizasyona gerek kalmamaktadır. Kısa mesafeli bir yol bulunamadığında, robota ait yerel yörünge planlayıcısı, problemi global yörünge planlayıcısına ileterek hedefe alternatif bir yol bulunup bulunamayacağının incelenmesini ister.

Yörünge Planlama Teknikleri:

1. Konfigürasyon-Uzayı Metodu:

Bir parçanın konfigürasyonu, parça üzerindeki her noktanın konumunu bütünüyle belirleyen parametreler kümesidir, konfigürasyon-uzayı ise tüm olası konfigürasyonların kümesidir. Konfigürasyon-uzayı metodunda, mobil robotun hareketinin engeller arasından planlanması problemi, mobil robotun noktasal olarak temsil edilerek genişletilmiş konfigürasyon-uzayı engelleri arasından hareketi problemine dönüştürülür. Engellerin genişletilmiş olmasının nedeni, mobil robotun noktasal olarak gösteriminden kaynaklanmaktadır. Engellerin gerçek boyutları öyle arttırılmaktadır ki, genişletilmiş engellerin kenarları, mobil robotun kenarlara çarpmadan ilerleyeceği yörüngeyi temsil etmelidir.

2. Serbestyol Metodu:

Konfigürasyon-uzayı yaklaşımına bir alternatifte, yörünge planlama problemini serbest alanı doğrudan araştırarak çözmektir. Serbestyol metodunda, serbest alanın temsili üstüste bindirilmiş genelleştirilmiş koniler vasıtasıyla gerçekleştirilmektedir. Mobil robotun öteleme hareketi oluşturulmuş serbest yol boyunca olurken, dönme hareketleri ise oluşturulmuş serbest yolların kesişme noktalarında gerçekleştirilmektedir.

3. Genelleştirilmiş Voronoi Diagramları Metodu:

Serbestyol metodunun daha rafine hali, Genellleştirilmiş Voronoi diagramlarından yararlanarak serbest alanın açık bir şekilde temsili ile gerçekleştirilebilir. Geneleştirilmiş Voronoi diagramları, her noktası aynı zamanda iki veya daha fazla engele yakın olan serbest alanın bir alt kümesi olarak tanımlanmaktadır. Kenarlar, engeller arasındaki maksimum açıklığı temsil etmektedir ve böylelikle de robotun geçmesi için en emin yolu göstermektedir. Engellerin dışındaki alan sürekli olarak diagram üzerinde deforme edilebilir. Serbest alan içindeki iki nokta arasında bir yörünge bulmak için, her nokta için diagram üzerinde bir yol bulmak ve daha sonra bu noktaları diagram üzerindeki yolları izleyerekten birleştirmek yeterlidir.

4. Hücresel Ayırma Metodu:

Konfigürasyon-uzayı, mevcut alanı hücrelere ayırma yolu ile araştırılabilir. Hücrelerin boş, dolu veya karışık olmasına göre konfigürasyon-uzayı engellerinin mevcut olup olmadığına karar verilir. Hücre eğer tamamiyle engel tarafindan doldurulmuşsa dolu olarak, hiçbir engelin mevcut olmaması halinde boş olarak, engelin bir kısmı tarafindan doldurulmuş bir kısmı boş ise karışık olarak adlandırılır. Genellikle boş hücrelere karşılık gelen düğüm noktaları ile bir 'bitişiklik grafiği' oluşturulur. Daha sonraki adımda, grafik heuristik bir arama algoritması ile araştırılır. Arama algoritması olarak genellikle A* algoritması veya modifiye edilmiş bir versiyonu kullanılır. Bu işlem, boş hücreler arasında bir yol bulana kadar devam eder.

5. Hipotez Kur ve Test Et Metodu:

Hipotez kur ve test et metodu, otomatik yörünge oluşturma problemini çözmek için gerçekleştirilen ilk metoddur. Bu metodda, başlangıç noktası ile hedef noktayı birleştiren bir yörünge önerilir. Daha sonraki adımda yörünge engellerle çarpışmaya karşı test edilir. Eğer engeller mevcutsa, çarpışmaya neden olan engeller incelenerek farklı bir yörünge önerilir.

6. Potansiyel Alan Metodu:

Yöntem hedefin robota sanal bir çekici kuvvet, yolu üzerindeki engellerinde aynı biçimde sanal bir itici kuvvet uygulaması gerektiği biçiminde bir senaryoya dayalıdır. Hedefe yönlendirilen çekici kuvvetle, karşılaşılan engellerin uyguladığı itici kuvvetlerin bileşkesi olan R vektörü, robotun bulunduğu pozisyona göre hesaplanır. R vektörünün büyüklüğüne ve yönüne göre robotu harekete geçiren hızlandırıcı kuvvet ve robotun alacağı yeni pozisyon hesaplanır. Bu işlem robot hedefine varana kadar tekrarlanır. Potansiyel alan metodlarında ortak nokta, çevrenin önceden bildirilen bir koordinat sistemine sahip olması, bu koordinat sistemi içinde engelleri temsil eden geometrik şekillerin verilmesi ve robotun bu verilerle hedefine giderken izlleyeceği yolun durağan hesaplanmasıdır. Potansiyel alan metodlarından biri de Sanal Kuvvet Alanı yöntemidir. Sanal Kuvvet Alanı Yöntemi gerçek zamanda belirlenebilen engellerden kaçınan yolların saptanmasına olanak vermektedir. Robot bu yöntemle yoluna çıkan beklenmedik engellerin arasından hedefine hızlı, sürekli ve düzgün bir hareketle ulaşabilir. Yöntemde robotun karşılaştığı engellerin önünde durması gerekmemektedir.

Mobil Robotlarda, kontrol yapısında bulunması gereken başlıca özellikler aşağıdaki maddeler halinde özetlenebilir:

1. Programlanabilirlik:

Programlanabilirlikten kastedilen, kullanışlı bir robotun çeşitli görevleri hardware veya software donanımının değiştirilmesine ihtiyaç duymadan yerine getirebilmesidir. Yerine getireceği görevler, tüm detayların verilmesine ihtiyaç duyulmayacak düzeyde verilebilmeli ve o anki bulunduğu statü ve çevreye göre robot bunları yeterli bir şekilde yerine getirebilmelidir.

2. Gürbüzlük:

Robot tüm hareketleri için tek bir alt sisteme dayanmamalıdır. Mümkün olabilecek ve gerekli tüm algılayıcı ve fonksiyonlardan yararlanabilmelidir.

3. Tutarlı davranış:

Robotun davranışının amacıyla ters düşmemesi gerekmektedir. Çevresine karşı tepkilerinin bu amaçlar doğrultusunda yönlendirilmesi gerekir.

4. Kontrol yapısının özerk, adaptif ve reaktif olması:

Bu üç kavram birbiriyle bağlantılıdır. Çevre koşulları değişkendir, tamamiyle bilinmemekte ve zamanla değişmektedir. Bu nedenle robotun özerk olması gerekmektedir.. Özerklikten kastedilen nokta, robotun görevlerinin kendisi tarafindan yerine getirebilmesi ve çeşitli alt sistemleri idare edebilmesidir. Kontrol yapısı adaptif olmalıdır çünkü robot, mevcut durumlara göre davranışlarında değişiklik yapabilmelidir. Bu da duruma göre davranışı üretme veya seçmeyi gerektirmektedir. Kontrol yapısının reaktif olması ise olayları zamanında algılama ve göreve göre tepki verme yeteneklerini gerektirmektedir.

Yukarıda belirtilen özellikleri gözöününde bulunduran özerk mobil robotlara ait kontrol yapıları iki sınıfta toplanmaktadır.

1. Kontrol Yapısının Fonksiyonel Ayrılması:

Fonksiyonel ayrılmada mobil robot kontrol yapısı acenta olarak bilinen her biri belirli bir fonksiyonu yerine getiren bir grup işlemden meydana gelmektedir. Acentalar genellikle robotun Algılama sistemi, Planlama sistemi, Gezinim sistemi ve Kontrolöründen meydana gelmektedir. Bu yapıya sahip özerk mobil robot kontrol sistemlerinden,

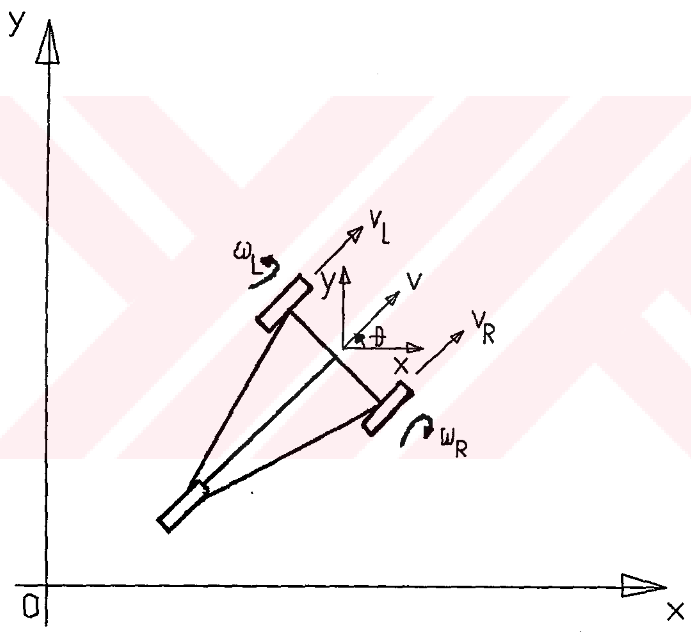
Kavramalı Kontrol Yapısı
Yayılı Kontrol Yapısı
Hilare Kontrol Yapısı

tez kapsamında incelenmiştir.

2. Kontrol Yapısının Davranışsal Ayrılması:

Kontrol yapısının davranışsal ayrılımında ise mobil robot çeşitli davranış düzeylerinden meydana gelen bir yapı sergilemektedir. Davranış düzeylerine robotun engellerden kaçınması, hedefsiz bir şekilde dolaşımı, çevreyi keşfetmesi v.b. örnek olarak verilebilir. Bu tip kontrol yapısında tüm sistemde meydana gelebilecek bir hata yerine sadece robotun performansını indirgeyecek bir yapı söz konusudur. Ayrıca kontrol yapısına başka davranış tabakaları da sonradan eklenebilir. İncelenen Tabakalı Kontrol Yapısı davranışsal ayrılmaya açıklayıcı bir misal teşkil etmektedir.

Mobil robotlarda hareket kontrolü problemine bir örnek oluşturmak amacıyla, Bölüm 6'da 3 tekerlekli mobil bir robot platformun noktadan noktaya hareket kontrolünün bilgisayarda küçük bir simülasyonuna yer verilmektedir. Şekil 1'den görüldüğü üzere, aracın ön kısmında iki tekerlek ve arka kısmında bir tekerlek mevcuttur.



Şekil 1. Mobil robotun konfigürasyonu

Ön tekerleklerin herbiri bağımsız olarak kontrol edilen DC motorlar tarafından tahrik edilmektedir. Arka tekerlek, aracın dengesini destekleyen serbest bir tekerlektir. Aracın yön sevki, iki ön tekerlek arasındaki hız farkından yararlanılarak elde edilmektedir. Diferansiyel tahrikin anlamı, ön tekerleklerden birine diğerine nazaran daha fazla moment verilmesidir bu şekilde araç dar alanlarda yüksek manevra kabiliyeti mevcut, kendi ekseni etrafında sıfır dönme yarıçapına sahip olarak dönebilmektedir. Aracın konumu ve oryantasyonu düzlemsel haraket söz konusu olduğundan üç parametre (x, y, θ) ile temsil edilmektedir. Aracın kinematik ve dinamik denklemleri elde edildikten sonra durum-uzay denklemleri oluşturularak durum-uzay değişkenleri olarak sırasıyla aracın kütle merkezinin çizgisel hızı V, açısal hızı $\dot{\theta}$ ve oryantasyonu (yön açısı) θ seçilmiştir. Mobil robotun simülasyonu elde edilen durum-uzay denklemlerinin, Runge Kutta IV sayısal integrasyon yöntemi

kullanılarak, bilgisayarda Matlab 4.0 mühendislik paket programlama diliyle çözümü sonucu gerçekleştirilmiştir.

Mobil robotun bir noktadan bir noktaya hareket kontrolü, aracın oryantasyon (yön açısı) ve çizgisel hız kontrolü vasıtasıyla gerçekleştirilmiştir. Oryantasyon kontrolünde PD kontrol algoritması, çizgisel hız kontrolünde ise PI kontrol algoritması kullanılmaktadır. Araç ara hedef noktalarına durmadan ulaşmakta en son hedef noktaya geldiğinde yavaşlayaraktan durmaktadır. Aracın bir noktaya farklı başlangıç oryantasyonunda hareketini ve çoklu noktaları içeren yörüngeye ait hareketlerini gösteren simülasyonlarda, durum-uzay değişkenlerinin ve sisteme girişlerin simülasyon zamanı süresince değişimleri grafik olarak çizdirilmiştir.

CHAPTER 1. INTRODUCTION

1.1. Introduction to Mobile Robot Concept

A mobile robot may be defined as a freely programmable or autonomous vehicle which can automatically move according to its mobility which may be defined as the ability to move in a given direction [1]. Looked at the most general way, a mobile robot is a machine which can move as a whole in a controlled way with some degree of autonomy. Most industrial robots are mounted on a fixed base and can not move around. Their work envelope is thus limited to the volume reached by the motion of the robot's joints. Hence, these robots must have their work brought to them. A mobile robot, however, can go where the work is and can transport items between fixed-base robots, thereby opening up many new application fields [2]. A mobile robot may have many of the same features such as manipulators, vision systems, and specialized sensors, etc., as fixed-base industrial robots, but at least three factors make them unique, all of which are due to requirements imposed by the robot's mobility. The mobile robot needs :

1. A locomotion system to allow it to move,
2. Steering capability,
3. A mapping function to determine where it is, a goal planning function to determine where it wants to go and a collision-avoidance system. It will also need an autonomous navigation capability.

As design considerations, for stability the robot needs a low center of gravity, but it must be high enough to clear expected obstacles. Another concern is motive power. The robot must have sufficient on-board battery power to operate for long periods without having its batteries recharged. But longer running times require more battery capacity, which adds weight to the robot. All of this additional weight could reduce the robot's payload-carrying ability. The longer the robot goes without a recharge, the longer it will take to charge it when needed, increasing the robot's down time. These considerations have led to a conceptual design for a mobile robot consisting of three sections:

1. A base that provides mobility. In most mobile robots, in order to keep the center of gravity low, the base usually holds the drive mechanism, the locomotion and steering systems, servo control and power amplification, and a source of mobile power.
2. A midsection that provides intelligence, control, and sensors. Sensors supply necessary collision-avoidance and navigation data and handle special-purpose requirements imposed by the application.
3. A working section that performs the robot's application-oriented functions. The working section may be completely passive, or it may have an arm that interacts with its environment. Some mobile robots have small robot manipulators installed to perform load and unload functions. According to these considerations, the typical structure of a wheeled indoor mobile robot is shown in Fig 1.1.

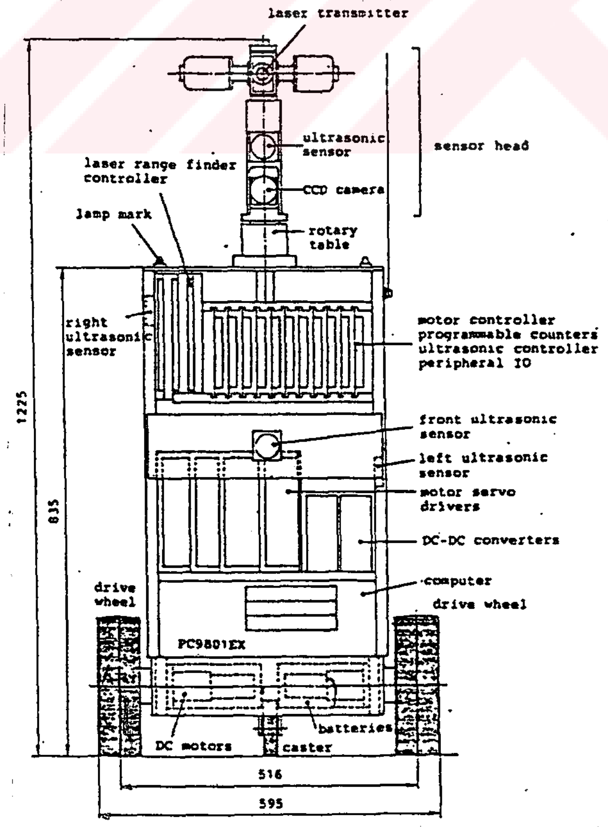


Figure 1.1. The structure of a mobile robot

The mobile mechanism of the robot is the conventional two-wheel-driven type with a caster wheel to support the body posture [3]. With this mechanism the robot is capable of moving straight, turning and pivoting. The drive wheels have pneumatic tyres and caster wheel has a solid tire. A DC servomotor with harmonic gear drives each main wheel. By the pulse drive system using an encoder , it is easy to control the rotational speed as well as the the rotation angle of the wheel. The same control systems are use for the caster wheel and for the rotary table of the sensor head. Intelligent motor control boards are used between the computer and the motor drive circuit boards. They make it easy to control the wheels for they reduce the processing load of the computer. The encoder signals of the main wheels are used to estimate the position of the robot by dead-reckoning. The robot has four ultrasonic sensors and a laser range sensor as external sensors. Three ultrasonic sensors are attached on the front, the right and the left sides of the body. The last ultrasonic sensor is mounted on the rotary table with the laser range sensor to get range information in any direction. The on-board computer is the combination of CPU 80286 and NDP 80287 of 10 Mhz. All the information processing, such as the actuators, sampling of the sensor data and communication with the host computer is done by this computer. The role of the host computer is to control the system and to interface the system with the operator. The host computer is the combination of CPU 80386 and NDP 80387 of 16 Mhz. This computer is linked with a mobile robot using a wireless modem and linked with the real-time motion measurement system using a serial line.

Mobility in robots has been a desirable goal for years, and it is becoming more important as robot technology continues to expand. The first step in mobility for industrial applications was the gantry robot built by Olivetti in 1975. The next improvement in mobility were wheeled vehicles. The first wheeled vehicles developed were teleoperated robots that had a remotely located operation to indirectly steer and control the robot's motion. Teleoperated robots have performed very well in hazardous environments, but because they require a human operator, they are too expensive for normal factory operations. After teleoperated robots, the next step was Automated Guided Vehicles (AGV). As it is seen from Fig.1.2, these models are like unmanned forklift trucks. They can handle large loads from one are of the factory to the another, but usually constrained to follow a fixed path.

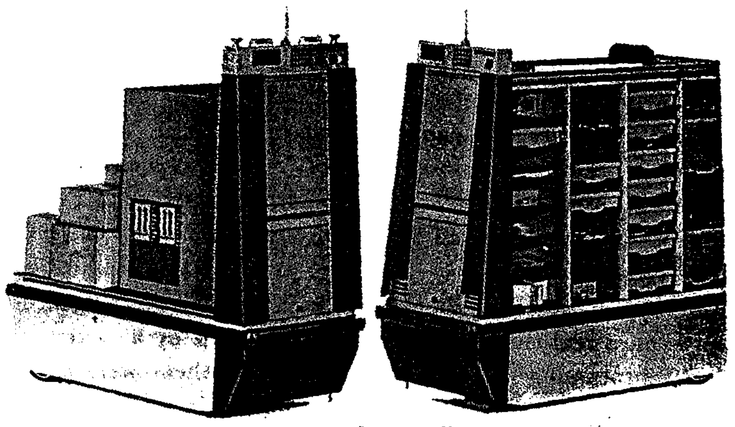


Figure 1.2. Bell and Howell Automated Guided Vehicles

The path is wire buried in the floor or a paint stripe on the floor that the robot is programmed to follow. AGVs employ a radio frequency sensor to follow a buried wire or an optical sensor to follow a visual path. In wire guided systems, low power rf signals are carried by wires imbedded in the floor. The AGV system interprets amplitude or phase variations of this signal to determine when it is going off-course. A servo loop controlling the AGV to steering system brings an off-course AGV back on center over the wire. Wire guided systems have a higher installation expense and provide less flexibility for system changes, but generally need less maintenance than a photooptical system. In photooptical system, either paint or a reflective tape is used for path control and an on-board photoelectric tracking system detects the difference between the desired path and the floor. Maintenance for optical path-based systems usually requires restriping about every six months. Looking toward the year 2000, major changes will have an impact on the intelligence of AGVs encompassing both on-board and off-board control systems [4]. These changes rely on advances in microcomputer technology and in the software coding industry. It is possible for most current configurations of AGVs to perform off-guide path maneuvers. There are three constraints which limit the current configuration of AGVs. The first concern is to bring computer processing power down to a price that is cost beneficial to the end user. Today's computers do have limitations, especially in processing power and speed. The second concern is the limitation of current software. The tremendous cost of writing or coding for an intelligent AGV system today offsets the cost justification of a system. The third limiting factor is the precision tracking capability of an AGV while off the guide path. Today, this is commonly handled by mounting an encoder on one of the AGV wheels to determine distance traveled and angularity of the wheel for location from the guide path. Developing better tracking and error correction systems will give AGVs precise tracking capabilities. In the next years, the control for the AGVs will be more closely involved for decision-making purposes. Artificial intelligence will increase the sophistication of the AGV systems. Along with this development will come decision-making capabilities on board the AGV. The AGV will be able to determine what functions it is to do next by deriving the answer from a number of different circumstances or inputs. From these inputs, the AGV will be able to determine speed, location, and best route.

In fact most desirable type of mobile robot is completely autonomous. As a class of robotic system, autonomous mobile robots constitute one of the important steps in the evolution of robotic intelligence and structure. At this step, a mechanical system oriented to perform a specific job is given an ability to move in space by some conventional means such as wheels, legs, tracks, and to provide this motion independently on human interference. The property of "autonomous" is understood as the ability to independently make intelligent decisions as the situation changes [5]. Such an ability is possible if intelligence allows a certain level of independence. For example, if the general object of motion is formulated by a human operator, but the specifics of the particular motion are taken care of by the robot with no direct human involvement. Therefore, different degree of autonomy can be considered. Robots can be generally controlled by a human operator. However, some of the operations can be planned, controlled, and executed with no human participation, they are left to the robot's decision. Autonomous mobile robots must cope with dynamically changing and unpredictable real-world environments. Hence, they must rely on more extensively on sensor-derived information than on precompiled world models and canned action sequences [6]. Briefly, the requirements for the autonomous mobile robots are as follows:

1. The robot should be able to formulate and generate explicit action plans using an implicit goal or mission description as input.
2. The robot should be able to execute and supervise explicit action plans independently.
3. Understanding of its world and to derive coherences with the current action.
4. Communication and interaction with its world.
5. Capability to react to unforeseen events or situations
6. Capability of active or passive learning for skill acquisition or internal model building.

These requirements yield to systems which may plan, execute and monitor autonomously .

1.2 A Brief Overview of Mobile Robot Projects

Early projects in mobile robotics date back to the beginning of the 1970's. In general terms, the major emphasis was on problem areas and approaches derived from the artificial intelligence domain. Researchers studied about representations for world modeling and issues in planning and problem solving. As a result, the complexity of problems in sensing, perception, and control was vastly underestimated and the systems built did not perform as originally expected and no substantial contributions were made concerning robotics and computer vision issues that today are more directly associated with mobile robot research. During the latter part of the 1970's, a lower level of research ensued. At the beginning of the 1980's, however, the studies began to increase in mobile robots, because of the advances in processing power and sensor systems enabled a new round of experiments in mobile robots to be conducted. Another reason, other robotics and sensing technologies advanced to the point that practical applications in restricted domains become feasible.

Brief historical overview of research projects conducted in the mobile robotics domain can be summarized as follows:

- **Shakey :** Shakey is one of the first mobile robots [7]. It was developed at the Stanford Research Institute, Stanford University, California, as a part of a project from 1966 through 1972. Shakey had a TV camera, an optical range finder, and several touch sensors. It operated in a highly constrained artificial environment populated with large blocks, and used a precompiled map of its surroundings. The project led to significant developments in development of logic-based planning and problem-solving techniques but did not focus on sensing or real-world modeling issues, thereby producing a system with extremely limited performance.

- **Jason :** Jason was developed at the university of California, Berkeley. It had an on-board computer, as well as ultrasound range sensors and infrared proximity detectors. A substantial part of Jason's computation was done on board, but experimentation with it seems to have been largely confined to tele-operation work.

- **The JPL Rover :** In the early 1970's, NASA in cooperation with Jet Propulsion Laboratory (JPL) initiated a program oriented to reduce ground support requirements, provide real-time control, new mission opportunities, and improve reliability and performance in support of space exploration, space assembly, automation of manufacturing facilities, launch, and earth orbital operations, and man-controlled remote operations of systems in hostile environments. In its "robot demonstrated program" (1973) JPL aimed to achieve the autonomous goal-directed coordination of locomotion, manipulation, perception, and cognition in a close to reality environments diagram to develop planetary explorers. [8] Its first prototype was a semi-autonomous vehicle equipped with a laser range finder, a stereo pair of TV cameras, and a manipulator arm.

- **The Stanford Cart :** The Stanford Cart was built at the Stanford Artificial Intelligence Laboratory, Stanford University, California. The Cart was a simple remotely controlled mobile platform equipped with a camera mounted on a slider mechanism, and a radio transmitter. It used sparse stereo vision for obstacle mapping by tracking sets of points over multiple steps. The milestone project emphasized low-level vision and operation in real-world environments .

- **Hilare :** The Hilare robot was built at the Laboratoire d'Automatique et d'Analyse des Systémes, CNRS, Toulouse, France. It is equipped with ultrasound sensors, a camera, and a triangulation-based laser range system. Research has concentrated on multisensory perception, planning, and navigation in maps represented as graphs.

- **Vesa :** The Vesa robot was developed at the Laboratoire d'Applications des Techniques Electroniques Avancées, of the Institut National des Sciences Appliquées, INRIA, Rennes, France. It has a bumper skirt, ultrasound range finders, and an on-board manipulator. Issues in multisensing, mapping, and autonomous path planning and navigation have been investigated.

- **The Yamabiko Robots :** The Yamabiko series of mobile robots was developed by Kanayama and Yuta at the Institute of Information Science and Electronics, University of Tsukuba, Japan [9] . One of several Japanese research efforts in autonomous mobile robots, The Yamabiko project has developed several mobile robots and explored issues in mobile robot programming, path planning, and navigation .

- **The MRL Robots :** The MRL robots were developed at Carneige Mellon University's Mobile Robot Lab, Pittsburgh, Pennsylvania. Research has addressed various problems in perception, navigation, and control for autonomous mobile robots. Specific projects have included the design, construction, and operation of several mobile vehicles; formulation of control strategies for wheeled mobile robots; research in sparse and dense stereo vision; range-based mapping, path planning, and the investigation of high-level planning and control issues.

- **The Autonomous Land Vehicle Project :** It was sponsored by the Defense Advanced Research Projects Agency of the Department of Defense. The autonomous land vehicle project is a multiyear effort that concentrated on road following and rough terrain traversal, using detailed stored maps and relying on sensor data for simple local navigation and obstacle avoidance.

- **The Hermies Robots :** The Hermies series of robots were developed at the Oak Ridge National Laboratory's Center for Engineering Systems Advanced Research, Oak Ridge, Tennessee [10]. They incorporate sonar and vision sensors, manipulators mounted on a mobile platform, and on-board processing, and have been used for research in visual perception, goal recognition, navigation in unknown dynamic environments, and the development of task-oriented manipulation strategies.

- **The VaMoRs Robot :** The VaMoRs autonomous vehicle project was developed at the Universitat der Bundeswehr, Germany. The project has developed an integrated spatiotemporal approach to automatic visual guidance of autonomous vehicles.

- **The KAMRO Robot :** The KAMRO project [11] was developed at the Institute for Real-Time Computer Systems and Robotics at the University of Karlsruhe, Germany. It was focused on the development of a mobile two-armed robot for experiments in autonomous navigation, docking, and assembly.

- **The FRC Robots :** The Field Robotics Center, at the Robotics Institute, Carneige Mellon University, has concentrated on the development of tele-operated and semi-autonomous robots for large-scale applications, including inspection and decommissioning of nuclear facilities, operation in hazardous environments, mining, and road following.

- **Blanche :** The Blanche mobile robot is an experimental device developed at the AT&T Bell Labs, Princeton, New Jersey [12]. It is designed to operate autonomously within structured environments, including offices and factory areas. It uses an optical range finder, and has been used for experiments in robotic programming languages, sensor integration, and techniques for error detection and recovery .

• **The Stanford Mobi :** The Stanford Mobi [13] is an omnidirectional platform equipped with a stereo camera system and sonar sensors. It has been used for indoor navigation. Issues in sensor integration, uncertainty handling, recovery of geometric world models, and their interaction with precompiled world maps have been investigated.

• **The MIT Mobots :** Several widely different vehicles have been developed at MIT's Artificial Intelligence Laboratory, Cambridge, Massachusetts, to investigate robust low-level sensing and control mechanisms, and, their use in layered control architectures.

• **The LSU-RRL :** Research at the LSU Robotics Research Laboratory, Baton Rouge, Louisiana concentrates on key areas, including intelligent sensing and navigation, exploring different methods of sensing and navigation by autonomous mobile robots in unknown environments.

The ARL Robots : The autonomous Robotics Laboratory at the IBM T.J. Watson Research Center, Yorktown Heights, New York, has focused on developing robots with agile and robust sensing, navigation, control mechanisms, and on developing automatic robotic learning strategies.

1.3 Application Areas of Mobile Robots

1.3.1 In Healthcare:

It has become increasingly obvious that there are many medical applications for present and future robotic systems. In the area of rehabilitation the Japanese "Meldog" guide robot (Fig.1.3) for the blind is undergoing pre-production tests 14].

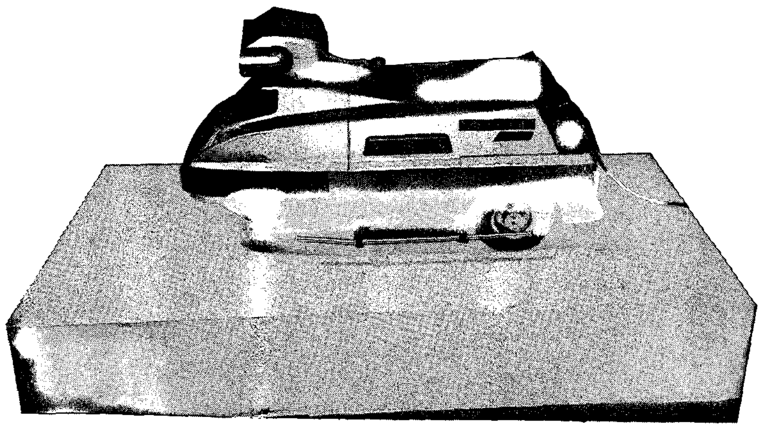


Figure 1.3. Meldog guide robot

This is a powerful mobile system with a comprehensive computer memory: It carries in its memory a detailed map of the town, intelligent sensors equipped to recognize street junctions, signpost, and other geographical landmarks. The robot monitors its progress and provides the necessary navigational instructions to its blind user. Thus the blind person is instructed by the robot to stop or to turn right or left. These control commands are transmitted by control switches on the harness. The person need not worry about his or her walking speed because Meldog constantly makes the necessary adjustments.

As another example, the hospital transport mobile robot, HelpMate [15], developed by Transition Research Corporation is expected to become an answer to the current shortage of help in hospitals . Specifically, it addresses the need for assistance with such tasks as point to point delivery. Examples of delivery tasks include meal trays, pharmacy, and lab supplies, patient records and mail. Navigational sensors include dead reckoning, optical vision, sonar, infrared sensors and structured light to detect obstacles in its travels. A second vision system watches for regular arrays of ceiling lights. Obstacle detection and avoidance algorithms allow it to adjust to its environment. Given the unstructured environment of a typical hospital, one technical objective of the HelpMate project is to successfully handle the uncertainty present and to navigate based only upon as little prior knowledge as possible.

1.3.2 In Military:

When the Gulf War began in January 1991, the world was soon told that the computer-based systems had a large part to play. The Gulf conflict was to a large extent planned and prosecuted by robot warmakers. In the early 1980's, the American Company Robot Defense Systems conceived the fully automated PROWLER (Programmable Robot Observer With Logical Enemy Response), a vehicle the size of a tank and able to perform a wide range of tasks under computer control [16]. The PROWLER designers hoped that before the turn of the century, the mobile system would be able to carry out a number of tasks, including sentry duty, reconnaissance, clearing minefields, laying mine, transporting of ammunition of other supplies to battlefield units, testing for chemical and biological agents and radioactivity, collecting human casualties, attacking armor, and defending from air attack. Many of the design features are now proven in practice. The system computer is well equipped to detect movement and body heat in a battlefield environment, and the associated techniques of pattern recognition, data collection by sensors and high-level decision-making by computer are well established in industrial applications. A different class of military mobile robot is the Ohio State University's six-legged Adaptive Suspension Vehicle, shown in Fig.1.4 , designed for ground too rough even for tracked vehicles.

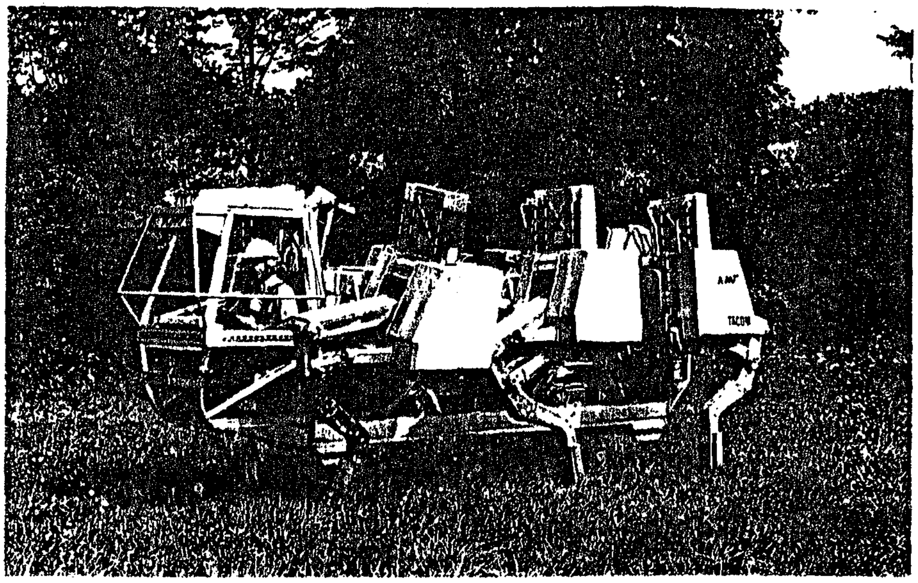


Figure 1.4. Adaptive Suspension Vehicle

In Britain, there is also a research program called Mobile Autonomous Intelligent Device (MAID) . Research is regarded as falling into four areas: Mobility, sensing, effectors such as grippers, and intelligence, which includes knowledge representation, mission planning, navigation and route finding among obstacles [17].

1.3.3 In Prisons and for Security Purposes:

It is clear that many military robot systems can be adapted for use in a civil environment, as effective guards in prisons and as security devices for the company sites and their locations. Already robot guards are in operation, albeit in a largely prototype capacity, in some parts of the United States. Denning Mobile Robotics of Massachusetts has agreed to supply the Southern Steel Company, the nation's largest manufacturer of detention equipment, with several hundred robots a year for several years. The aim was to provide systems that could be used to augment human guards. For example, to patrol prison corridors at night. The robots are designed to move at about 3 km/h to detect human odours, and to transmit sound and pictures.

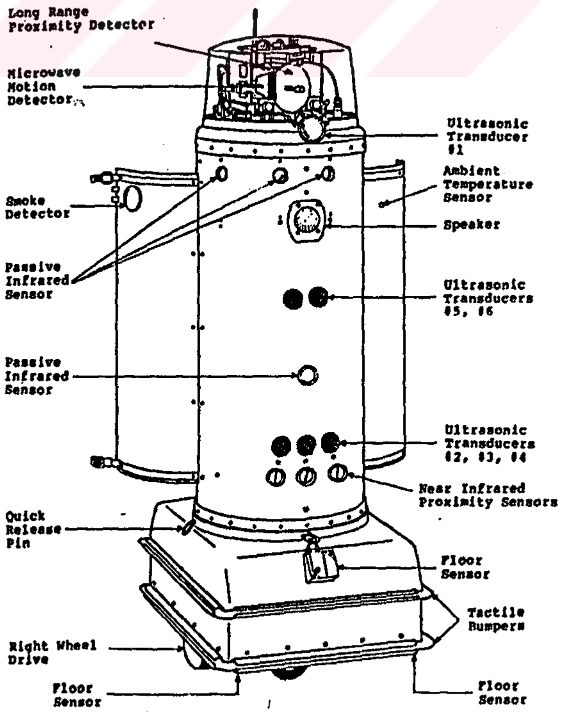


Figure 1.5. Robart-II

Such robots would be able to undertake suicide missions in the case of prison riots, venturing into hazardous situations and transmitting information for as long as they survive. The Robart project , already encountered as a military system, also has a relevance to civil security requirements. The structure of Robart-II is shown in Fig. 1.5. There are now three Robart systems progressively equipped with ever more sophisticated technologies.

1.3.4 In Hazardous Environments :

High radiation environments present many hazards to human beings; operators can wear protective clothing, but there are still many potential dangers, particularly when lengthy repairs have to be carried out to a reactor or when an entire plant has to be decommissioned. Already various types of robot systems have been used in nuclear power plants for maintenance and decommissioning purposes and their use is expected to increase in the years ahead. For example, the Rover I remote reconnaissance vehicle was employed to penetrate the radioactive basement of the damaged Three Mile Island Nuclear Plant in The USA, and to take pictures and radiation measurements. A sibling vehicle, Rover II, then took material samples from the inside walls of containment buildings, so aiding the creation of a comprehensive database of damage information. Another vehicle, Workhorse, has been developed at Carneige Mellon University in Pittsburgh to take a range of operational initiatives in a nuclear plant environment. Workhorse comes equipped with on-board saws, scrapers, and wrenches, all of which can be selected by a 7m mobile arm.

In the same vein, a group of nine Japanese companies, supported by the Ministry of International Trade and Industry (MITI), is working to develop a family of robots for dismantling obsolete nuclear plants. The German company Kerntechnische Hilfdiest Gesellschaft (KGH), has supplied a mobile robot to aid and surveillance and sampling tasks at Chernobyl.

Mobile robots are also potentially attractive for inspecting dangerous areas in fires and for providing a fire-proof vehicle can be sent through a fire to rescue people trapped by flames or smoke. An alternative is for the robot to carry supplies such as escape devices, breathing equipment or a portable refuge to trapped people. Such a robot would probably be teleoperated by radio or, if the building was designed with this in mind, could be a wire-guided AGV. A third possibility is wall-following navigation, this would need a map of the building to be prepared in advance.

1.3.5 In the Home:

Any self-respecting domestic robot must be able to perform many tasks such as cleaning, washing, cooking, childminding, tidying, bathing the baby. There are no robots that can do this much, and the ones that can do even a few of the necessary tasks have usually been dauntingly expensive. As early as 1976, Quasar Industries in New Jersey had designed and built a 1.6m tall robot that could be programmed to perform a number of domestic chores. For example, it could map floors, mow lawns, and run through simple cooking procedures. The aim was to make this robot available by 1980 at a cost around $ 4000, but after the initial launch not much was heard about it. By the early 1980's a number of companies in the USA and Japan were working to develop economical domestic robots. A product of the Androbot venture, TOPO looked like a friendly plastic toddler, though it lacked the average toddler's capacity for independent initiatives. A later Androbot product, BOB (Brains On Board) was intended to be a brighter fellow. Other robots of this type include Heath's HERO (Heath Education on Robot) able to detect intruders, to perform simple domestic tasks, and to converse in limited fashion with a human user.

There should be scope for task performance both inside and outside the house. For example, it may be useful, for a domestic robot to be able to clean all the outside windows of a house, to secure a displaced tile and to repair a damaged chimney. Such a robot might be able to climb the outside walls of a house using suction feet. Already a robot called the Skywasher has been designed to crawl on the surface of building while washing the windows. This device measures 1m by 1m and weights 20kg; it is equipped with six sets of suction cup feet to enable it to walk horizontally or vertically in a sidestep fashion, a set of wipers and a washing fluid system. Such a system shows what we may expect in the years to come. However, James Crowley from Carneige Mellon University in Pittsburgh says that "The technical state-of- the-art for a domestic robot can be compared to the situation in aviation at about the time of the Wright brothers' first manned flights."

1.3.6 In Entertainment:

Robots have also been designed to play instruments. One of the most remarkable robots involved in entertainment is WABOT-2, an impressively anthropomorphic keyboard playing fellow from Waseda University in Japan. WABOT-2 [18] has dexterous fingers to play an organ or a piano, feet to operate pedals, eyes to sigh-read sheet music and a brain to supervise all the perceptual and motor processes. WABOT-2 can walk, hear, and converse with human beings .

1.3.7 In Planetary Exploration :

Active exploration of other planets could answer many questions about the nature and origins of our solar system. sending astronauts or remotely controlled vehicles is possibility, but a manned expedition is highly unlikely in the near future, and conventional teleoperation is impractical because of the long signal times. A more promising approach is to launch an unmanned prospector and a vehicle to return collected samples to earth.

In the 1970's, there was a Mars rover research program at the Jet Propulsion Laboratory sponsored by NASA, and the results of the research were reported at the time [19]. However, the program ended in 1979. Recently, interest in planetary rovers has revived. The broad objectives of NASA's proposed Mars Rover and Sample Return mission would be to observe and gather materials representative of the planet's geophysical, meteorological, and biological conditions and return a varied selection of samples. Since the payload of the return vehicle is limited, the mission requires a sophisticated on-site system that can explore, assay, and select.

Current NASA planning is for a launch of a Mars rover and sample return mission perhaps in 1998, if funding is forthcoming. The JPL experimental rover can be seen in Fig. 1.6

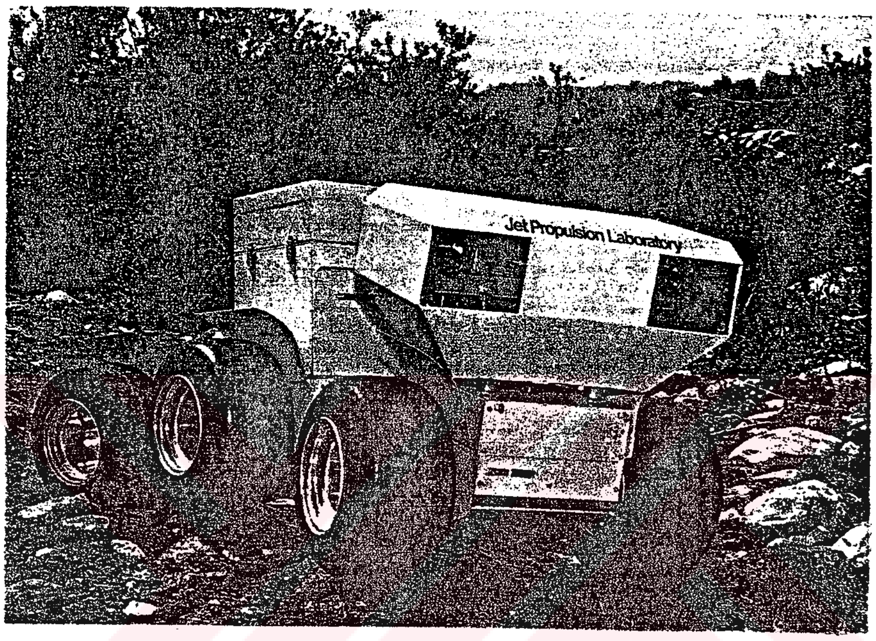


Figure 1.6. A Mars rover, JPL

Carneige Mellon University initiated a research program that addresses the central robotics challenges of designing a roving explorer capable of operating with minimal external guidance . The purposes of this research are to confront issues not faced by laboratory robots, to identify and formulate the difficult problems in autonomous exploration, and to generate insights, principles, and techniques for their solutions. According to these considerations, a prototype legged rover, called the Ambler (loosely an acronym for Autonomous MoBile Exploration Robot), and testing it on full-scale, rugged terrain of the sort that might be encountered on the Martian surface.

1.3.8. In Underwater Applications:

The underwater robots are generally used in visual inspection, cleaning work, non-destructive inspection work, simple maintenance work. The conceptual view of an underwater robot with a tether can be seen in Fig 1.7.

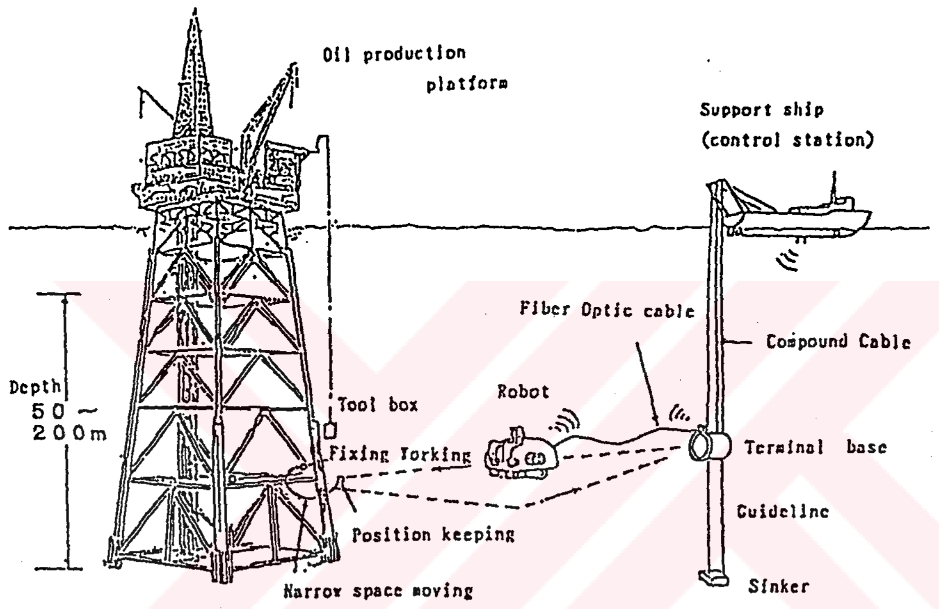


Figure 1.7. Underwater robot

Tetherless autonomous mobile robots are desirable for works at great depth since the drag imposed by the tetherline is an important limiting factor. As the cable length is increased, it becomes thicker ; then the power required to pull it increases correspondingly [20]. Because of the water murkiness, vision can be utilized only at extremely short ranges. The range of perception can be increased by using the ultrasonic systems. However the quality of perception drops substantially. One of the autonomous underwater mobile robot was developed at the University of New Hampshire. Their EAVE-East Vehicle is controlled by the on-board expert system. The master-computer also directs all other intelligent functions. An eight-legged mobile robot called RECUS, made by Komatsu, has been used for surveying the seabed in preparation for bridge construction and has been proposed for other tasks such as leveling mounds of rubble.

The Naval Ocean Systems Center has developed an unmanned free swimming submersible for underwater pipeline search and inspection. A multitasks operating system provides functions for pattern recognition, sensor and effector coordination, communication with the surface and representation of knowledge acquired from the environment. The swimmer could work either completely autonomously or tethered with a fiber optic link. The swimmer locates a pipe by searching for a magnetic signature similar to that of a pipe. An acoustic altimeter is also used to survey the ocean bottom near the pipe.

CHAPTER 2. LOCOMOTION SYSTEMS OF MOBILE ROBOTS

Locomotion is the subsystem that gives the robot its mobility. In the design of mobile robots, the form of locomotion utilized by the robot must first carefully selected. As this selection of the type of locomotion is crucial to the performance of the robot, poor selection can make success in the design of the mobile robot near to impossible. Locomotion systems can be classified as follows:

a. Wheels
b. Legs
c. Tracks
d. Articulated body
e. Combinations of the basic configurations

In theory, the choice depends upon trade-offs in power requirements, costs, desired locomotion characteristics, and the surfaces which the robot must transverse. In practice, however, the choice is usually wheels. Because there are several engineering problems remain unsolved in legged mobile robots. Wheels perform well if the terrain is level or has only a limited slope, is reasonably smooth and good traction . Otherwise tracks can be used, they are well suited for outdoor, off-road applications. They can even climb stairs. In wheeled and tracked locomotions [21], the configuration is said to be artificial, because in animals endless rotational motion is not observed at all.

Generally speaking, mobile robot based on leg is not practical. Its mechanism usually weights too much since large number of actuators are required to drive the system of the multi degree of freedom and at present commercially available actuators are bulky and heavy. To achieve the required motion of the vehicle some actuators may have to work in ways which do not contribute to propulsion, but waste some power. Legged mobile robots are difficult to design, the computations required for leg control and balance are numerous and difficult and their power drain is so large that these robots are currently seen as only experimental models in the university or R&D Laboratories. But the legged locomotion has special characteristics such as the greatest capability of going over most types of terrain and climbing stairs. A smooth ride can be achieved on rough ground; a leg does not like a wheel, waste power by always having to climb out of a rut of its own making and is less susceptible to digging deeper and deeper until the vehicle stops and feet may do less damage to the ground than tracks or wheels.

2.1. Wheeled Mobile Robots:

Although legged and tracked locomotion systems has been studied, the overwhelming majority of the mobile robots which have been built and evaluated utilize wheels for locomotion. Wheeled mobile robots are more energy efficient than legged or tracked robots on hard, smooth surfaces. Three wheel types are used in Wheeled Mobile Robot (WMR) designs:

1. Conventional
2. Omnidirectional
3. Ball Wheels

In addition, conventional wheels are often mounted on a steering link to provide an additional DOF [22]. The configurations of the three wheels are shown in Fig 2.1

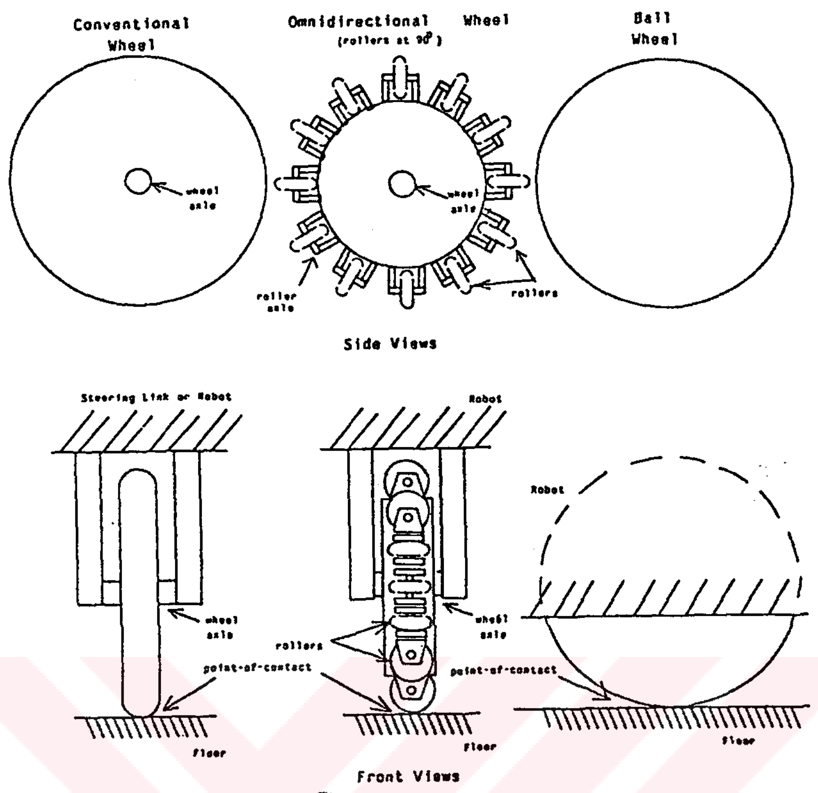


Figure 2.1 Wheel types

The DOFs of each wheel are indicated by the arrows in the Fig 2.2.

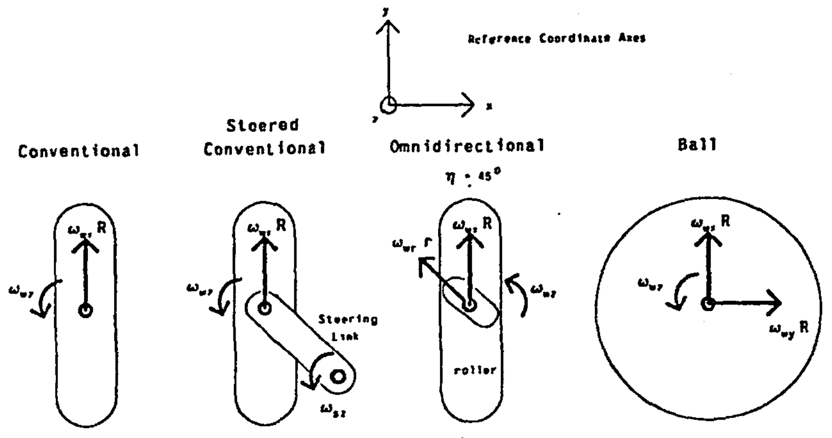


Figure 2.2. Wheel DOFs

The kinematic relationships between the angular velocity of the wheel and its linear velocity along the surface of travel are compiled in Table 2.1.

Table 2.1. Wheel equations of motion

Kinematic Relationships		
Conventional and Steered Conventional	**Omnidirectional**	**Ball**
$V_y = \omega_{wx} R$	$V_y = \omega_{wx} R - \omega_{wr} r \cos(\eta)$	$V_y = \omega_{wx} R$
$V_x = 0$	$V_x = \omega_{wr} r \sin(\eta)$	$V_x = \omega_{wy} R$
$\omega_z = \omega_{wz}$	$\omega_z = \omega_{wz}$	$\omega_z = \omega_{wz}$

where,

V_x and V_y	:	x and y components of the linear velocity of the wheel at the point of contact.
ω_z	:	z component of the angular velocity of the wheel at the point of contact.
ω_{wr}	:	Angular velocities of the roller about their axles.
ω_{wx}, ω_{wy}, ω_{wz}	:	x, y, and z angular velocities of the wheel about its center.
ω_{sz}	:	Angular velocity of the steering link about the steering axis.
η	:	Angle of the roller axles with respect to the wheel orientation.
R and r	:	Radii of a wheel and roller.

The conventional wheel having two DOFs is the simplest to construct. It allows travel along a surface in the direction of the wheel orientation and rotation about the point of contact between the wheel and the floor. The rotational degree of freedom is slippage, since the point of contact is not stationary with respect to the floor surface. Even though the rotational slip is defined as a DOF, slip transverse is not considered to the wheel orientation a DOF, because the magnitude of force required for the transverse motion is much larger than that for rotational slip. The conventional wheel is by far the most widely used wheel; automobiles, roller skates, and bicycles utilize this wheel. The omnidirectional wheel has 3 DOFs. One DOF is in the direction of the wheel orientation. The second DOF is provided by motion of rollers mounted around the periphery of the main wheel. In principle, the roller axles can be mounted at any nonzero angle η with respect to the wheel orientation. The omnidirectional wheel in Fig. 2.3 has roller axle angles of 45° respectively. The third DOF is rotational slip about the point of contact.

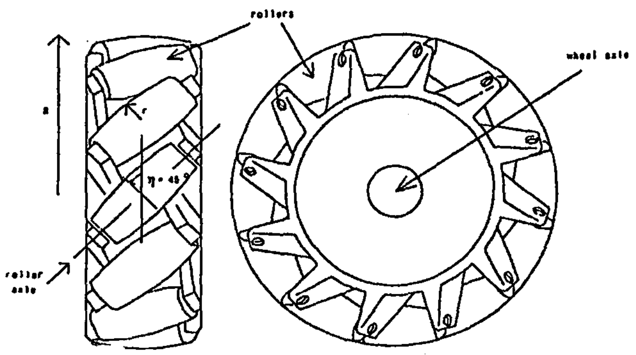


Figure 2.3 Omnidirectional wheel

It is possible, but not common, to actuate the rollers of an omnidirectional wheel with a complex driving arrangement. The most maneuverable wheel is a ball which has three DOFs without slip. An omnidirectional wheel which is steered about its point of contact is kinematically equivalent to a ball wheel, and may be a practical design alternative. A definition for a wheeled mobile robot can be given as follows:

"A wheeled mobile robot is a robot capable of locomotion on a surface solely through the actuation of wheel assemblies mounted on the robot and in contact with the surface." A wheel assembly is a device which provides or allows relative motion its mount and a surface on which it is intended to have a single point of rolling contact. Each wheel and all the links between the robot body and the wheel constitute a wheel assembly.

Kinematic models of WMRs are inherently different from those of stationary robotic manipulators and legged or tracked mobile robots. There are generally six practical assumptions to make the modeling problem tractable:

1. The wheeled mobile robot does not contain flexible parts. It states that the dynamics of such WMR components as flexible suspension mechanisms and tires are negligible. This assumption is made to apply rigid body mechanics to kinematic

modeling. Flexible structures may play a significant role in the kinematic analysis of WMRs.

2. There is zero or one steering link per wheel. This assumption limits the complexity of the kinematic model.

3. All steering axes are perpendicular to the surface. With this assumption all motions are reduced to a plane. Therefore, all components of motions are constrained to a rotation about the normal to the surface, and two translations in a plane parallel to the surface.

4. The WMR moves on a planar surface. This assumption neglects irregularities in the actual surface upon which a WMR travels. Even though this assumption restricts the range of practical applications, rough, bumpy, or rock surfaces do not lend themselves to energy-efficient wheeled vehicle travel.

5. The translational friction at the point of contact between a wheel and the surface is large enough so that no translational slip may occur. It ensures the applicability of the theoretical kinematic properties of a wheel in rolling contact for the two translational degrees of freedom. This assumption is realistic for dry surfaces, as demonstrated by the success of breaking mechanisms on automobiles.

6. The rotational friction at the point of contact between a wheel and the surface is small enough so that rotational slip may occur. The wheels must rotate about their points of contact to navigate a turn. Since WMRs also rely on rotational wheel slip, this assumption can be included.

Wheeled mobile robots usually have three or four wheels, although six or more are possible. Some configurations even have sets of double wheels. Four wheels offer more stability, if all are driven more traction, and slightly better clearance capabilities. Three wheels offer less complex steering, the elimination of shock absorption mounting, less weight for the wheel assembly and some navigational improvements. There are three fundamental ways of steering a wheeled mobile robot as it is shown in Fig. 2.4.

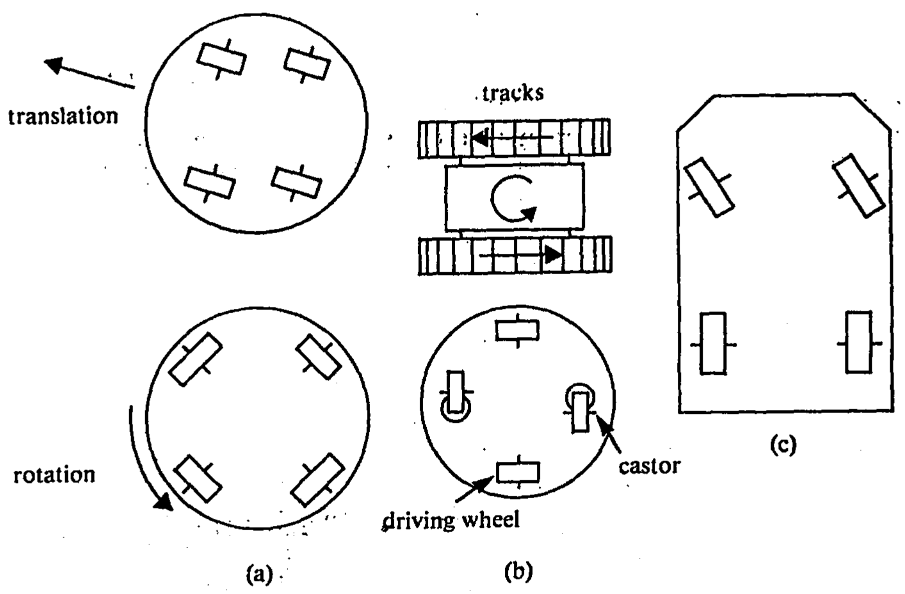


Figure 2.4. Steering methods for wheeled mobile robots

In the least restricted way, Fig 2.4-a, all the wheels can be turned together or differentially, resulting in translational or rotational motion or a combination. Such a vehicle can rotate about its own center and being very maneuverable, is favored for robots in buildings [17]. The second way used by some wheeled mobile robots, is differential steering of a pair of fixed wheels Fig 2.4-b. Again it allows rotation on the spot, but the vehicle can not move sideways. The third way, (Fig 2.4-c), is to pivot one wheel or a group of wheels, while others remain fixed; this the method used by cars and bicycles. It has rather poor maneuverability, but simplifies the construction of fast and efficient vehicles. This arrangement is rarely found in robots as turned round in a confined space or passing through a narrow gap tends to need complex back-and-forth maneuvering . Various wheel configurations can be also seen in Fig 2.5. Fig 2.5-a shows a configuration with two rear drive wheels and single front steering wheel. Fig 2.5-b shows all wheels providing drive and steering. This design has been used in Cybermation mobile robot. Another clever approach is the wheel chosen for TOPO, a personal home robot developed by Androbot. This technique shown in Fig 2.5-c has the wheels at an angle to the floor, making the robot easier to steer and more stable.

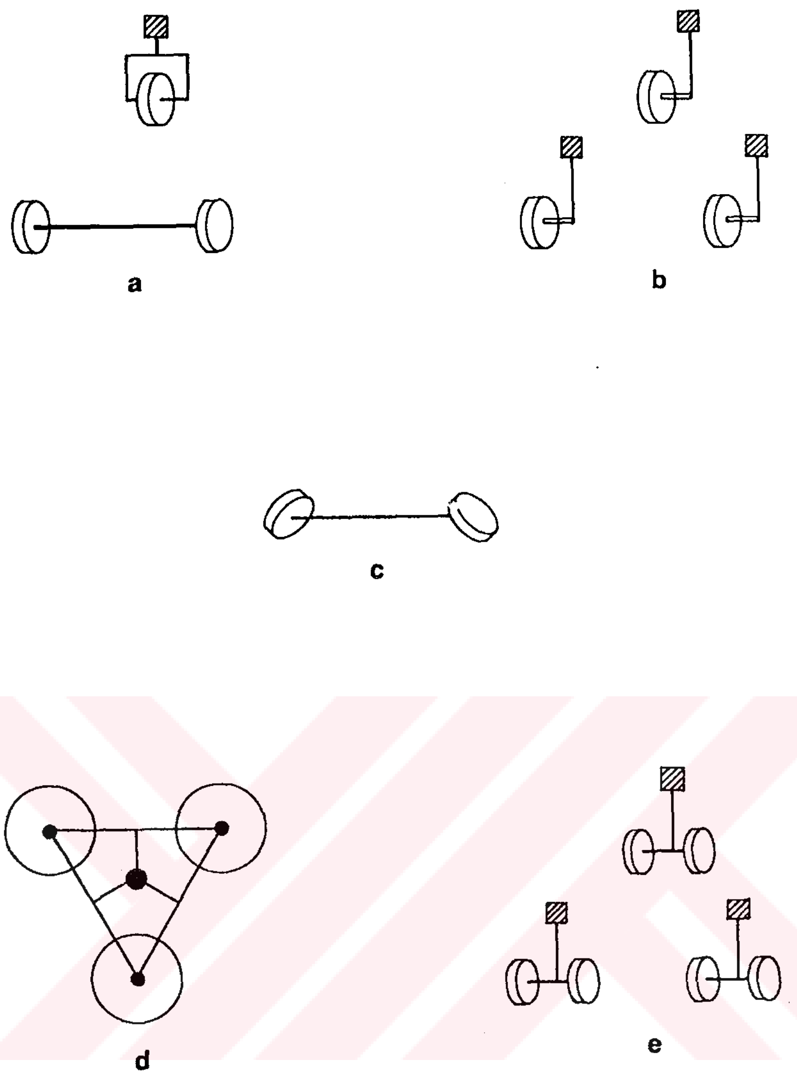


Figure 2.5. Various wheeled mobile robot configurations

Fig 2.5-d shows Weinstein's approach in which three rotating wheels spaced 120° apart allow the robot to climb stairs. The robot is normally supported by two of the wheels. When it comes to stairs or other obstacles of less than half the wheel's height, contact forces make the configuration rotate automatically and allow the robot to climb the stairs [2]. Carnegie Mellon's Pluto robot has two wheels on each of the support legs, as shown in Fig 2.5-e.

As a design consideration in wheeled mobile robots, larger wheels can roll over minor obstacles and allow the robot's undercarriage to clear small objects on the floor, but larger wheels raise the robot's center of gravity, which makes it less stable and contributes to the reduction of the robot's available dead-reckoning positioning accuracy. In other words, when a wheel goes over an object, the apparent path length increases, which affects the robot's internal position calculations. Because of stability and accuracy considerations, most systems have adopted small wheels from 10cm to 30cm.

Briefly, although wheels are limited in their terrain adaptability, they are extremely effective on even ground and the mechanisms can be simple and lightweight.

2.2. Tracked Mobile Robots:

Tracked vehicle design has been around for many years, such as on tanks and bulldozers. The two tracks can be bands or belts and have continuous or linked element arrangements. Driving wheels may be smooth or use sprockets. Some wheels are driven and others, called idlers, are allowed to freewheel. Some idlers support the vehicle weight as it runs along the ground, are then called rollers. Fig 2.6 shows the arrangements between drive wheels, idlers and rollers.

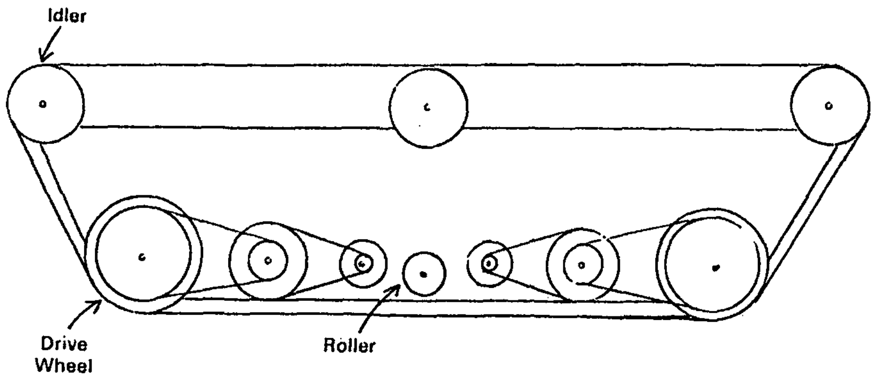


Figure 2.6. Tracked vehicle drive

For the tracked mobile robots to climb obstacles, the general track shape must be trapezoidal. Thus, the robot has its own built-in ramp. The height of the ramp must be greater than any obstacle the robot is expected to cross. The distance between tracks affects vehicle maneuverability as does track length. For good maneuverability, the width must be about 70% of the track ground contact length. In addition to maneuverability considerations, the length of any tracked vehicle is a function of the width of any unsupported area that the vehicle must traverse. As a minimum, the track length must be at least the length between the supports and proper design would use three times the unsupported area.

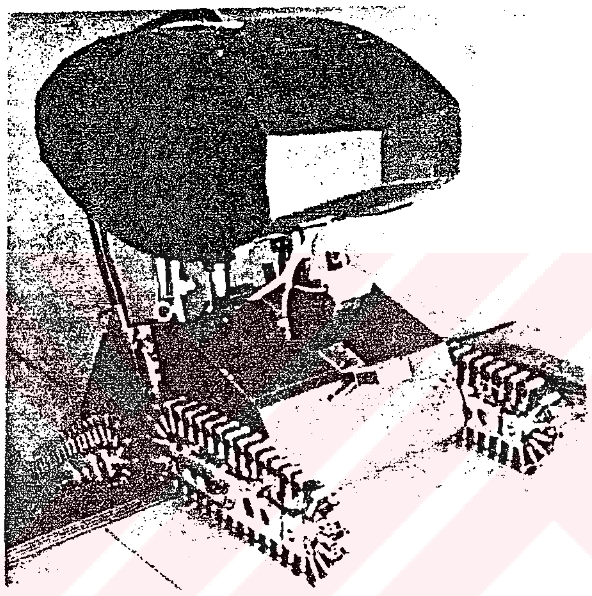


Figure 2.7 HELIOS 2

Being lower, tracked vehicles have more clearance problems than do wheeled vehicles. They can also cause greater floor damage. Although the track distributes loads over a greater area, they can still damage wires they may run over by pinching them. Total weight of the mechanism is much greater than for wheeled models and drive efficiency is poorer, so the vehicle uses much more power.

HELIOS 2, shown in Fig. 2.7, is an example of tracked mobile robot. It has developed at Tokyo Institute of Technology. It has four crawler tracks, each of which has an axled at the central and is torque controlled. The mechanism to drive the upper body to the pitching, and a pitching angle sensor are installed in HELIOS 2. By using these drive-sensor mechanism, the center of gravity of the entire vehicle is always maintained at the middle of the crawler while keeping the posture of the body horizontal when moving on a slope or staircase. The rolling angle is also sensed and body is maintained horizontal by the bending angle control of the tracks.

2.3. Legged Mobile Robots:

There are two serious reasons for exploring Legged Mobile Robots (LMR). One reason legs provide better mobility in rough terrain is that they can use isolated footholds that optimize support and traction, whereas a wheel requires a continuous path of support. As a consequence, a legged system is free to choose the best footholds in the reachable terrain whereas a wheel is forced to negotiate the worst terrain [23].

Another advantage of legs is that they provide an active suspension that decouples the path of the body from the path of the feet. The payload is free to travel smoothly despite pronounced variations in the terrain. A legged system can also step over obstacles . The performance of legged vehicles can, to a great extent, be independent of the detailed roughness of the ground. Legged mobile robots will need systems that control joint motions, sequence of the use of legs, monitor and manipulate balance, generate motions to use known footholds, sense the terrain to find good footholds and calculate negotiable foothold sequences. Most of these tasks are not well understood yet, but research is underway. A second reason for exploring legged mobile robots is to understand how humans and animals use their legs for locomotion.

The milestones in the development of legged robots can be seen in Table 2.2

Table 2.2. Development of Legged Robots

1850	Chebyshev	Designs linkage used in early walking mechanisms
1872	Muybridge	Uses stop-motion photography to document running animals
1893	Rygg	Patents human-powered mechanical horse
1945	Wallace	Patents hopping tank with reaction wheels that provide stability
1961	Space General	Eight-legged kinematic machine walks in outdoor terrain.
1963	Cannon,Higdon and Schaefer	Control system balances single, double, and limber inverted pendulums
1968	Frank and McGhee	Simple digital logic control walking of Phony Pony
1968	Mosher	GE quadruped truck climbs railroad ties under control of human driver
1969	Bucyrus-Erie Co.	Big Muskie, a 15.000-ton walking dragline is used for strip mining. it moves in soft terrain at a speed of 900 ft/hr
1977	McGhee	Digital coordinate coordinates leg motionsof hexapod walking machine.
1977	Gurfinkel	Hybrid computer controls hexapod walker in USSR
1977	McMahon and Greene	Human runners set new speed records on tuned track at Harvard. Its compliance is adjusted to mechanics of human leg.
1980	Hirose and Umetani	Quadruped machine climbs stairs and climbs over obstacles using simple sensors. The leg mechanism simplifies control.
1980	Kato	Hydraulic biped walks with quasidynamic gait.
1980	Matsuoka	Mechanism balances in the plane while hopping on one leg.
1981	Miura and Shimoyama	Walking biped balances actively in three-dimensional space
1983	Sutherland	Hexapod carries human rider. Computer, hyraulics, and human share computing task.
1983	Odetics	Self-contained hexapod lifts and moves back end of pickup truck
1987	OSU	Three-ton self-contained hexapod carrying human diver travels at 5 mph and climbs over obstacle.

An interdependence exist between the mechanical design of legs and gaits that is fundamental to performance control. The sophistication in control must be matched by sophistication in mechanical design in order to produce a useful machine. Gate analysis can provide data on stability, performance, or curved or uneven terrain and dynamic interaction of components and power efficiency. Although an understanding of animal joints and gaits is now generally understood, researchers in mobile robotics have chosen to develop low cost, simple concepts that can be easily controlled. Fundamental to control of legged mobile robots is a understanding of gaits. A gait of an articulated living creature or a walking machine is the corporate motion of the legs, which can be defined as the time and location of the placing and lifting to each foot, coordinated with the motion of the body in its six degrees of freedom, in order to move the body from one place to another. Animals select various gaits depending on terrain and desired speed. Sensory data is required to select intelligently the optimum mode for the desired operational speed or terrain. Knowledge of gaits provides the designer with a base of which control algorithims for walking machines can be written. Two forms of gait analysis have been developed analytical and graphical methods. Of the two forms, graphical method provides a quick understanding of the geometrical representation of the gait and is the most useful for mechanical designers. Footfall formulas are an early modern technique for representing gaits. The animal is observed from above as it transverses from left to right. Foot contact with the ground is indicated by the black circles with the arrows indicating direction as it is shown in Fig. 2.8.

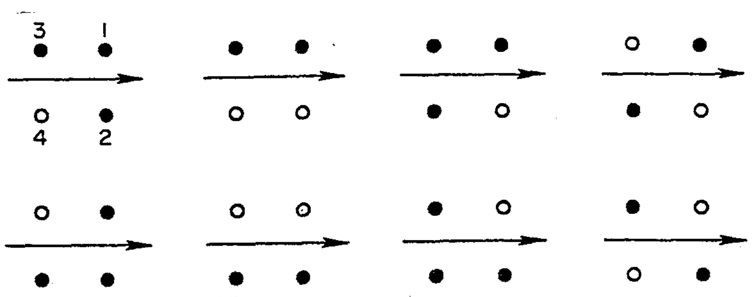


Figure 2.8. Footfall diagram for four legged animal

A more elaborate technique was developed by Hildebrand known as the "gait diagram". This is not only records the sequence placement and lifting of different feet, but also provides support and transfer phases of each leg as a function of time.(Read clockwise, the circumference of the circle represent the cycle time.) Other forms of gait diagrams include the successive support patterns of gait, with the arrow showing direction as it is shown in Fig. 2.9.

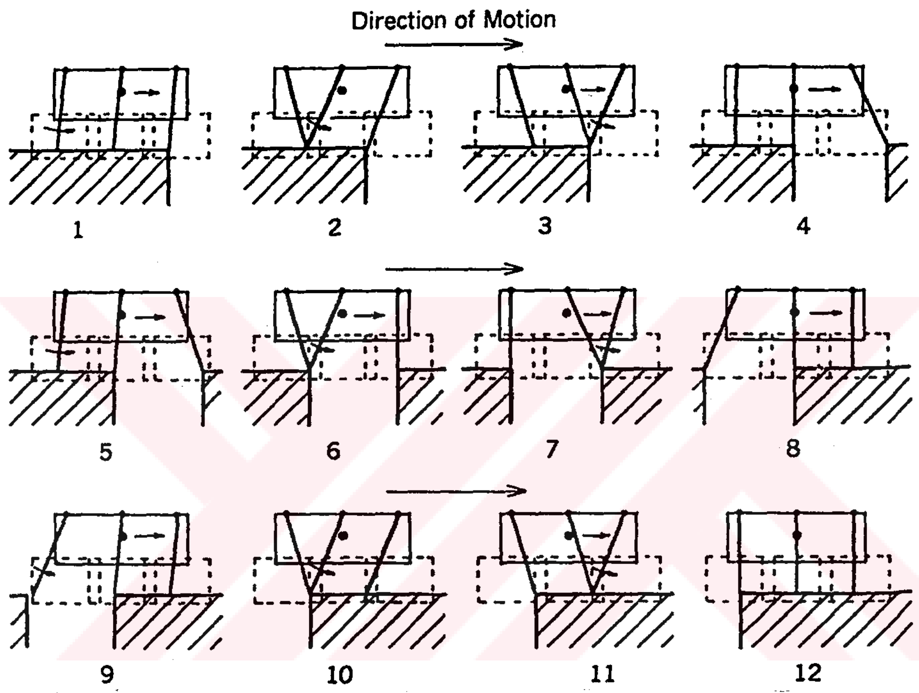


Figure 2.9. Successive Gait Pattern Diagram

The stationary gait pattern incorporate support patterns at different times and the corresponding body motion for each support pattern. It is useful for studying the gait stability margin of periodic gaits. The lateral motion sequence presents the machine in profile in a multiple sequence and provides center of gravity as an aid to understanding obstacle crossing. All of the above may be computer generated and are useful for building up more complex two-and three-dimensional graphics that provide walker stability and the gait adjustment necessary to traverse obstacles.

2.3.1. Types of Legged Mobile Robots:

A) One-legged Robots

A one legged mobile robot that could dynamically balance while hopping around was developed by Marc Raibert while at Carneigė Mellon University [24]. The main purpose was to develop the necessary algorithms for dynamic balance .

The equations of motion for a planar one-legged model, with a massless leg and hip located at the center of mass as shown in Fig 2.10 , are:

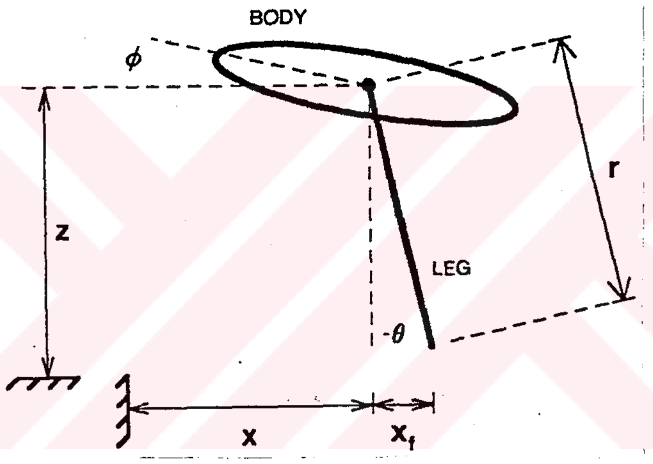


Figure 2.10. Model of planar one-legged system

$$m\ddot{x} = f\sin\theta - \frac{\tau}{r}\cos\theta \qquad (2.1)$$

$$m\ddot{z} = f\cos\theta + \frac{\tau}{r}\sin\theta - mg \qquad (2.2)$$

$$J\ddot{\phi} = \tau \qquad (2.3)$$

where,

x,z,ϕ	are the horizontal, vertical, and angular positions of the body,
r,θ	are the length and orientation of the leg,

τ	is the hip torque. Possitive τ accelerates the body in the positive ϕ direction,
f	is the axial leg force. Positive f accelerates the body away from the ground,
m	is the body mass,
J	is the body moment of inertia,
g	is the acceleration of gravity.

The unit used a remote computer to solve the motion equations, so a tether cable was required between robot and computer. This approach would automatically compensate for unbalancing forces, so even a hard shove against the side of the robot would not knock it over; the robot would just hop off in the direction in which it was pushed. This work is being continued at MIT.

B) Two-legged Robots

The Japanese have been leaders in two-legged walking robot research. Advantages of two-legged robots include narrower width and consequently greater maneuverability in confined spaces. WL-5, an early two-legged walker developed at Waseda University , took its first tentative steps in 1972. Hydraulicly powered, it uses disproportionately large feet for stability. A shuffler more than a walker, WL-5, achieved static walking. Invented by Ichiro Kuto at Waseda University, Tokyo, Japan,the design displays the first two evolutionary stages:

1. The oversimplified design deletes the knee and ankle joints,
2. Distributed actuators with standard motors or hydraulic actuators arrayed for powering the hip, knee, and ankle joints.

The latter stage is embodied in the WL-10RD (shown in Fig. 2.11) which started walking in 1984.

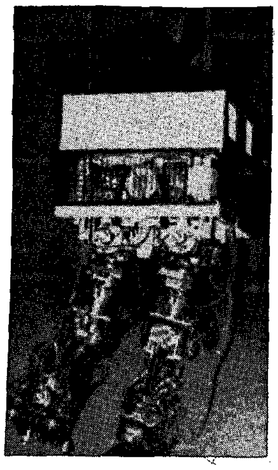


Figure 2.11. WL-10RD

It has six degrees-of-freedom per leg and walks forward, backward, and sideways, but can not turn. It takes 4.8 seconds for a stride of 45cm and weights 80 kg with a maximum step rate of 1.3 seconds per step. Its inventors describe the walking motion as taking place in two phases:

1. Change over phase, in which only the ankle joints of the legs move and the center of gravity is transferred from the rear leg to the front leg by moving the ankles only. In this phase, the robot, as an inverted pendulum, leans forward. This phase causes a large shock, which is absorbed by providing the ankles with flexibility.

2. Single support phase, in which the center of gravity remains almost unmoved, and the swing leg is moved from the backward position to the forward position.

WHL-11(Waseda Hitachi Leg) is a walking robot, with links resembling human limbs. The robot has six degrees-of-freedom per leg, compared with seven in humans. It walks slowly, taking 10s per step. With a step of 50.8 cm, the speed is thus limited to about 0.16 km/h. The robot is completely autonomous, containing its hydraulic pump, electric power, and computer system. Wl-12, the latest model in the series, succeeded the earlier models in 1987.

C) Four-legged Robots:

There has been less interest in four-legged robots because they are not as stable as six-legged ones, but they are more complex than two-legged robots. The MIT Quadruped [25], which is shown in Fig 2.12, is an example of dynamically stable running machine. Not limited to running, the Quadruped may also trot, pace, bound, pronk, and make transitions between these gaits.

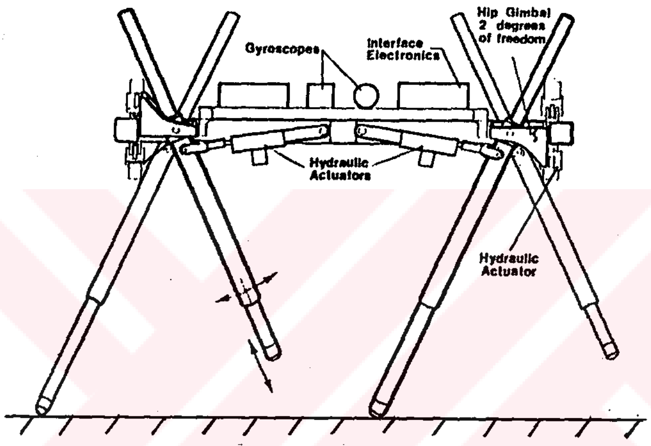


Figure 2.12 MIT Quadruped

An aluminum frame supports four opposing legs, hydraulic cylinders, hip, actuators, gyroscope, and computer interface electronics. Gimbaled hip actuators provide pitch and yaw and the telescoping legs perform the function of the extend and retraction of the knee. An air spring chamber in the end of each leg provides spring. Linear potentiometers provide sensing for leg length, overall leg length, leg spring length, and contact between the feet and the floor. The gyroscopes provide pitch, yaw, roll orientation of the body. Hydraulic power is provided through umbilicals.

Control algorithms were written for various trotting, pacing, and bounding gaits. Two level of control are used:

1. High level: Gait control for vertical bouncing motion, stabilizes the forward running speed, and keeps the body level.

2. Low level: It provides coordination of legs, manipulation of the legs for feet placement, relative forces a pair of legs exert on the ground, and torque the hips experience from the legs.

Leg structure of MIT Quadruped can be seen in Fig 2.13.

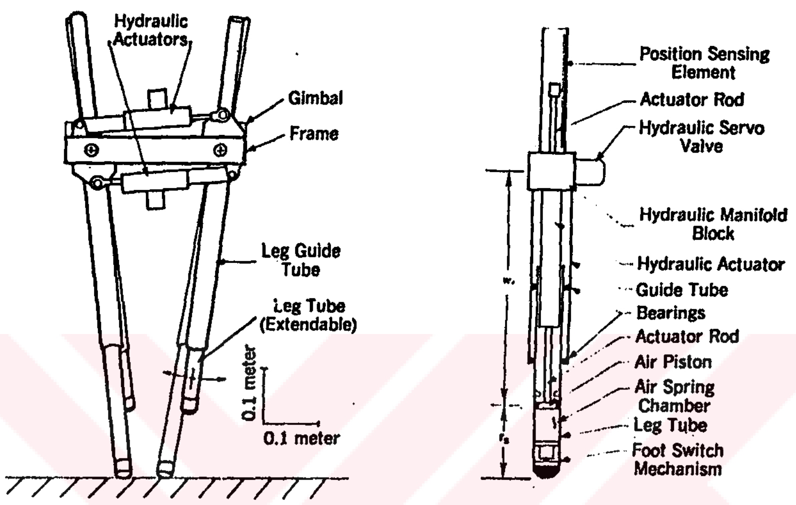


Figure 2.13 MIT Quadruped Leg Assembly

Another example of four-legged robot is TITAN IV which is exhibited at the science exhibition of Tsukuba in 1985. On sole of the feet are installed whisker sensors which identify the ground contact conditions and detecs the obstacles such as the step of a staircase 26]. The body of the vehicle is maintained horizontal by using inclination sensors. TITAN IV has made only statically stable locomotion with a velocity of 8 cm/sec at the exposition, but its control software has been improved to achieve dynamically stable walk.

D) Six-legged Robots

Ohio State University is building an ambitious machine , called the Adaptive Suspension Vehicle 27]. This multimillion dollar, DARPA funded project is an example of robotic 'Big Science". It is very interesting, but a person wanders if more could have been done to produce a practical machine. It must be used outdoors. Its dimensions are 5m long, 3.9m high, and 1.6m wide. It weights approximately 2600 kg, cruise speed is 2.25m/s, while top speed is 3.6m/s. A 900 cc, 70 kW motor cycle engine drives a 0.25 kW/hr flywheel that transfers power to a group of 18 variable displacement pumps interconnected via parallel line shafts that run the vehicle's length through toothed belts. The actuators are double acting hydraulic cylinders powered by the pumps, loading of each leg is obtained by reading the hydraulic pressures across the multiple actuators. It requires a driver, albeit with much less burden for the operator due to advanced computer control. The operator uses a six axes joysticks for general motion of the ASV with the control of the individual limbs and coordination of the legs controlled by computer. Sophisticated optical radar mounted on the top of the cab provides terrain information that is used for foothold selection and body orientation in some control modes. A vertical gyro provides a vertical reference, while rate gyroscopes and accelerometers provide information on body movements. All six legs are fitted with three axes force sensors to determine the three force components at the ankle. A pantograph structure is used on the legs Fig. 2.14.

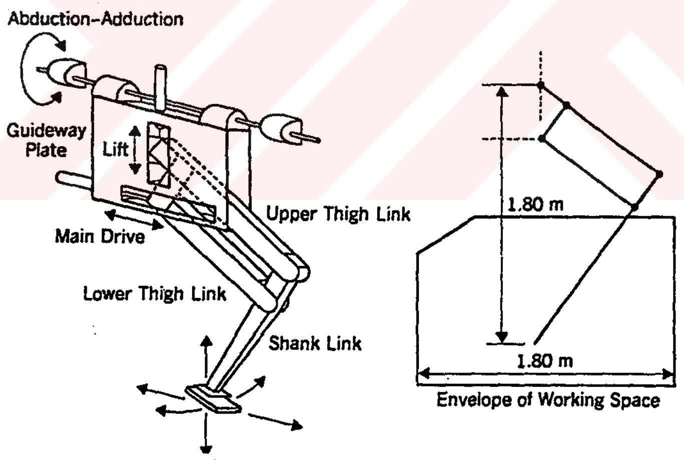


Figure 2.14 Pantograph structure of ASV Leg

Extention and retraction is powered by linear motion of the actuator. The upper portion provides lift, and abduction/adduction is generated by pivoting the entire mechanism. The hip achieves pitch and yaw in a very cumbersome way, that incorporates the knee into the hip mechanism with no ankle actuation.

One other famous six-legged walking robot is the FUNCTIONOID by Odetics, Inc. of Anaheim, California, has achieved perhaps the greatest fame of any modern robot system and has appeared on television and countless magazine covers as well as in numerous newspaper articles (shown in Fig 2.15).

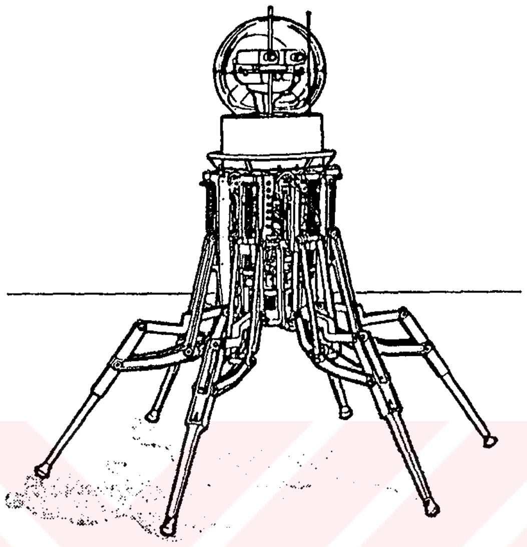


Figure 2.15 Functionoid created at Odetics

It weights 167.8 kg. A 24V, 25A/hr aircraft battery provide a self-contained power source with up to one hour of operation [28]. When walking 450 Watts of power are consumed, but at rest a mere two Watts are used. Eighteen motors are employed and the largest motors being used for lifting. On the reverse stroke, the motors act as generators, storing the surplus power in the battery. Brakes on the motors also help to conserve power. A double tripod, the system raises three articulated legs at a time to advance. The other three support the machine. The legs are a parallelogram design similar to the one used in industrial robot arms as it is shown in Fig 2.16. Ball screws operate the hip pitch and ankle pitch motion, and hip yaw swings the assembly via a gear drive for side to side motion. The robot can operate with a full load for about an hour before battery recharging is required. This device has a glass head where two cameras are housed.

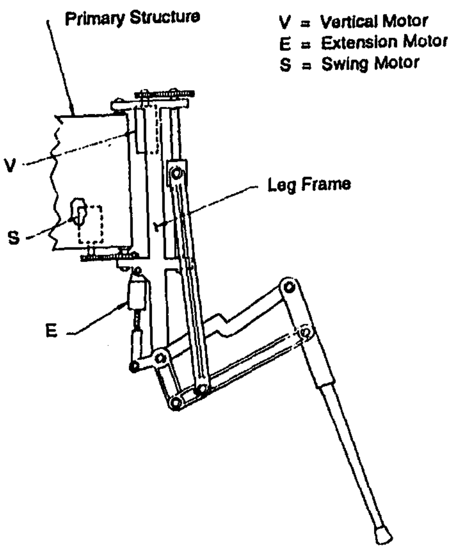


Figure 2.16 Functionoid Pantograph Leg

Another significant implementation is Ambler, built at Carneige Mellon University [29]. The aluminum Ambler was designed for planetary exploration. The Ambler configuration consists of six legs stacked coaxially at their shoulder joints as it seen in Fig 2.17.

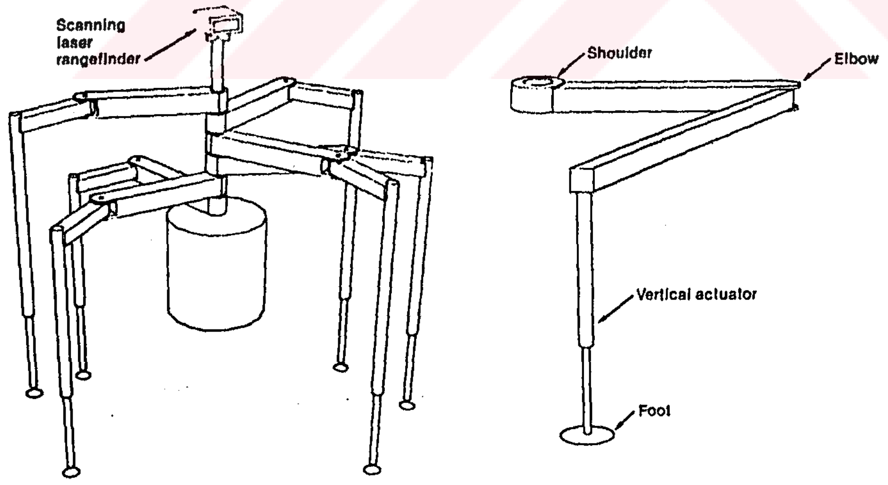


Figure 2.17 Configuration of Ambler

Each leg is mounted at different elevation on the central axis of the body and can rotate fully around the body. Each leg consists of two revolute joints that move in a horizontal plane to position the leg, and the prismatic joint at the end of the elbow link that effects a vertical telescoping motion to extend and retract the foot. Thus the locomotor has 18 degrees of freedom. The planar reach of a leg is 2.5 meters and the telescopic distances 1 to 2 meters, depending on the position of the leg on the stack. The average overall height of the Ambler is approximately 3.5 meters, and its nominal width is approximately 3 meters. With these dimensions, the Ambler can step over obstacles 1 meter high while maintaining a level body trajectory. Therefore the design provides a stable platform for sensors, scientific instrument and sample acquisition tool. The action of Ambler can be described as follows: A single leg reaches out in front of the other, places itself firmly on the ground like a ski pole and then pulls the machine forward. In the event of an accident or failure, any functional leg can reposition itself to substitute for any failed leg. Thus, The Ambler retains its ability to walk even if it loses mobility in two legs. Its two stacks are connected to an arched body structure that involves enclosures for power generation, electronics, computers, and scientific equipment.

2.3.2. Main Problems In Legged Mobile Robots

The legged mobile robots described above are not useful vehicles or even prototypes for such vehicles. They are experimental apparatus used in the laboratory to explore ideas about legged locomotion. Each was designed to isolate and examine a specific locomotion problem, while postponing and ignoring many other problems.

The problems remain to be solved before legged robots are transformed into practical machines that do useful work are as follows:

a) Terrain Sensing:

Perhaps the most important problem limiting current walking machines, as well as other forms of autonomous mobile robots, is their inability to perceive the shape and the structure of their immediate surroundings. Humans and animals use their eyes to locate good footholds, to avoid obstacles, to measure their own rate and direction of progress, and to navigate with respect to visible landmarks. The problem of giving machines the ability to see, has received intensive and consistent attention for the past thirty years. There has been steady progress during that period. Current machines can see well enough to operate in well-structured and partially structured environments, but it is difficult to operate autonomously in rough outdoor terrain.

b) Travel on rough terrain:

Complete knowledge of the geometry of the terrain, as might be supplied by vision or these other sensors, would not in itself solve the problem of walking or running on rough terrain. A system traveling over rough terrain needs to know or figure out what terrain shapes provide good foot holds, with sequence of footholds would permit traversal of the terrain and how to move so as to place the feet on the available footholds. It will be necessary to coordinate the dynamics of the vehicle with the dynamics of terrain.

There are several reasons that terrain becomes rough and therefore difficult to negotiate such as not level surface, limited traction, areas of poor or nonexistent support, vertical variations (minor or major), large obstacles between footholds, intricate footholds. The techniques that will allow legged systems to operate in these sorts of terrain will involve the mechanics of locomotion, kinematics, dynamics, geometric representation, spatial reasoning and planning. Limb control has been also a problem because three types of control are being attempted at once. These are dynamic balance, forward motion and turning. The computer programs for legged mobile robots perform several functions, including:

- sampling and filtering data from the sensors,
- transforming kinematic data between coordinate systems,
- executing, the three-part locomotion algorithms for dynamic balance, forward motion and turning,
- controlling the actuators,
- reading operator instructions,
- storing walking behavior.

In addition, design choices must be made in two other areas : The number of legs and the gait. In general, more legs means more balance, but control and coordination problems are more difficult. Gait refers to the order in which the legs are moved and to the number of legs on the ground at one time. Some gaits are more stable than others. The perception and control mechanisms required for legged systems to travel on rough terrain will require a substantial research effort, but the important problems can be solved within the next ten years if they are pursued vigorously.

c) Mechanical Design and System Integration:

When these sensing and control problems are solved, it will remain to develop mechanical designs that function with efficiency and reliability. Useful vehicles must carry their own power, control computers and a payload. There are several interesting problems such as energy efficiency, structural design, strength and weight of materials and efficient control. For instance, the development of materials and structures for efficient storage and recovery for energy will be particularly important for legged mobile robots.

d) Static and dynamic balance:

Static balance is balance with robot standing still. A robot, three or more legs, has static balance. A two-or one-legged robot must rely on dynamic balance or balance under motion, which is much harder to accomplish due to the generally high center of gravity of a legged robot. Even a three- or four-legged robot becomes potentially unstable as it moves its legs and requires dynamic balance to move. If three legs can always be kept on the ground at one time and the center of gravity of the robot is within the leg contact points, static balance is sufficient. This can be accomplished with many six-legged robots.

2.4. Articulated Body:

The articulated body mobile robot is a mobile robot composed of several articulated body segments linked linearly like a snake [30]. By using unified train-like configuration and the active coordinated motion of the segments, the articulated mobile robot posses following characteristics :

- It can pass over uneven terrain and narrow paths by actively adapting its long trunk to the ground topography.
- It can cross a ditch by stiffening the joint servomechanisms to bridge the ditch. At the same time, it can stablely wade a marsh by softening the join servomechanism to distribute its weight to all segments.
- The cross section of the path required for locomotion is small compared with the volume of the body. At the same time, the driving mechanism of each of the unit can be simply designed. Therefore the ratio of payload to the path cross section can be large.
- Because of the unified redundant structure, the reliability and maintainability can be high. The malfunctioning segment can be easily detached and displaced.
- It can easily be transported by breaking the robot down into individual segments.

As an example of articulated body mobile robot Koryu-II (KR-II) can be given. KR-II is a mobile robot composed of several articulated body segments linked linearly. Connecting mechanism between two units can be seen in Fig. 2.18.

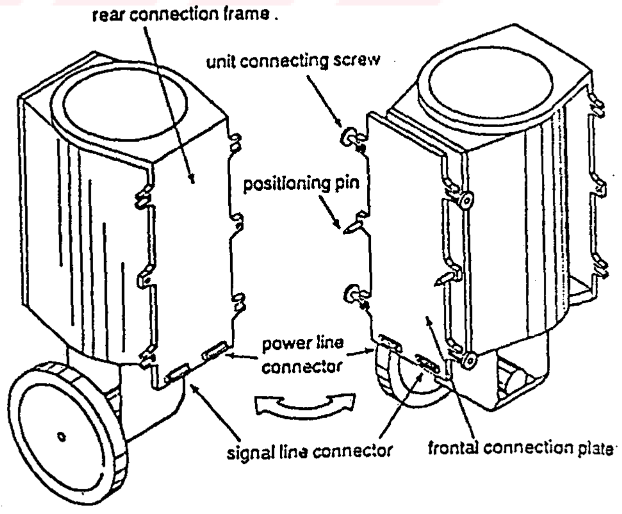


Figure 2.18 Unit Connecting mechanism of Koryu-II

It is constructed to make surveillance task for fire fighters at the fire site. Each segment of KR-II is generally cylindrical, and it is connected with its center axes standing up vertical to the ground. The linking of the articulation of KR-II is done by adjusting the alignment of connecting surface by a pair of pins, then fixed by six linking screws shown in Fig 2.18. There are two degrees of motion freedom at the joint of the segments. These are: The motion freedom to make vertical linear motion, and the motion freedom to make turning motion around the vertical axis of the segment. Also the propulsion is caused by the wheel attached on lower part of the segment. Thus, there are three degrees of freedoms on each articulation and the motion of KR-II is generated by the coordination control of these degrees of freedom.

2.5. Combinations of the Basic Configurations:

Some combinations of the basic configurations mentioned above can be also given as locomotion systems of mobile robots. The Mars Rover, four-legged vehicle with crawler attached at the heel, developed by JPL ; the five telescopic leg vehicle with a wheel attached at each heel developed by Hitachi Ltd. are the examples of this type of configuration [31].

CHAPTER 3. NAVIGATION SYSTEMS OF MOBILE ROBOTS

Navigation, from the Latin **navis**, a ship, plus **agere**, to direct, is the process of successfully directing the movement of a vehicle from one point to another. Navigation includes the processes of orientation in space, but only as a part of much broader operational activities. Navigation is the fundamental requirement of mobile robots. It is also defined as "the science of getting ships, aircraft, or spacecraft from place to place; the method of determining position, course, and distance traveled" in Webster's Ninth New Collegiate Dictionary. It is essential to know the whereabouts of the vehicle, although the accuracy of the estimate may vary according to the requirements of the vehicle's path. In the following, some navigation systems used in mobile robotics will be explained in details.

3.1. Dead Reckoning:

Odometry is an ancient method of recording distance. Recorded by Hero and Leonardo da Vinci, odometry, offers a low cost, reasonably accurate method of providing proprioceptive data by measuring the number of wheel revolutions and extrapolating distance from that data. This method by counting wheel revolutions, together with direction finding using a magnetometer or gyrocompass or differential wheel rotation, is often used for wheeled indoor robots. By measuring the distance traveled and the steering angle of the mobile robot, the position and heading of the mobile robots with respect to the start position can be determined. The easiest part of dead reckoning lies in distance traveled; if the robot continues in a straight line, no angle measurements are necessary. The distance traveled by the robot for each complete revolution of the drive wheels is:

$$X = 2\pi R \qquad (3.1)$$

R is the radius of the wheel.

The total distance traveled is given as follows:

$$X = 2\pi RN \qquad (3.2)$$

where, N is the number of wheel revolutions.

Although knowing the speed of the wheels and the time of travel allows to calculate travel distance, it is much more accurate to use wheel encoders to determine the number of revolutions and, from that, the distance traveled. After a turn is made by the vehicle, the mobile robot needs to know the actual angle of that turn. Fig 3.1 shows a method of determining the new coordinates (X_n,Y_n) of a mobile robot and the new heading angle θn.

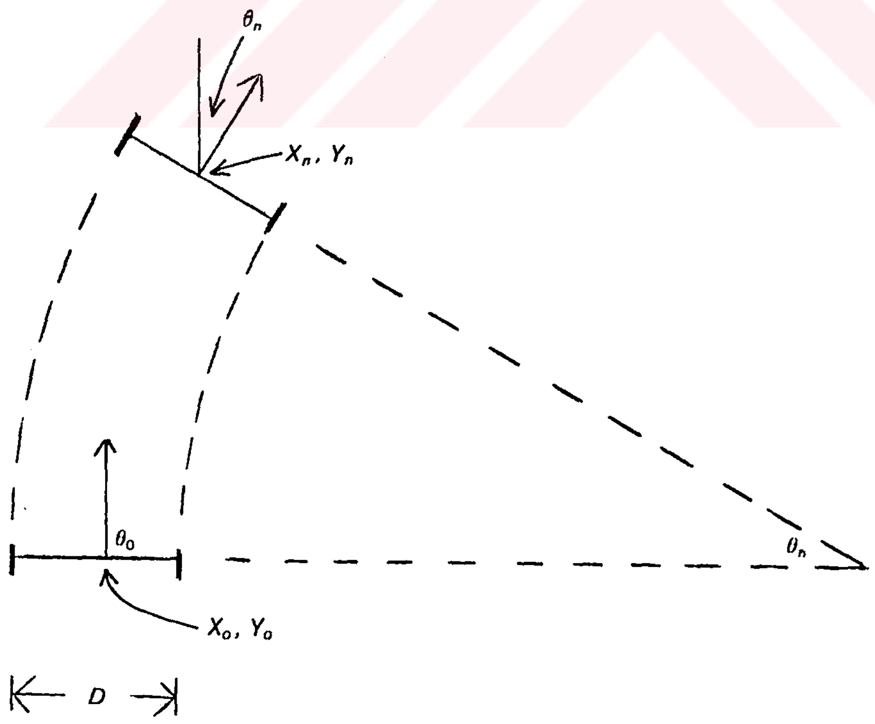


Figure 3.1. Turn calculations based on wheel encoders

It is based on the number of encoder counts, N_l and N_r, for the left and right wheels. Assuming both wheels have the same radius R:

Step 1: Compute the average number of counts:

$$N_A = \frac{N_l + N_r}{2} \tag{3.3}$$

Step 2: Compute the difference in count between sides:

$$N_D = N_l - N_r \tag{3.4}$$

According to the equation (3.4), if $N_D < 0$, then $\theta < 0$ and the robot turns left.

Step 3: Compute the change in the heading:

$$\Delta\theta = \frac{2\pi R N_D}{N_T D} \tag{3.5}$$

where, D is the linear distance between the wheels and N_T is the total number of encoder counts per wheel revolution. Distance traveled along the arc depends on the number of counts that occur for each revolution of the wheel and on the radius of the wheel.

Step 4: The new heading is

$$\theta_n = \theta_0 + \Delta\theta \tag{3.6}$$

Step 5: The new coordinates of the mobile robot,

$$X_n \approx X_0 + \frac{2\pi R N_A}{N_T}\cos(\theta_0 + \Delta\theta/2) \tag{3.7}$$

where, X_o is the previous value of X.

Step 6:

$$Y_n \approx Y_0 + \frac{2\pi R N_A}{N_T}\sin(\theta_0 + \Delta\theta/2) \qquad (3.8)$$

where, Y_o is the previous value of Y.

Odometry is simple and inexpensive, however, this method always shows a steadily increasing error due to wheel slippage and tyre elasticity[32]. Surface roughness and undulations may cause the distance to be overestimated, wheel slippage can cause the distance to be underestimated. Therefore the mobile robot needs to call at intervals at known locations at which the navigation system can be reset. Ideally, for a tricycle configuration, non-load bearing, knife-edge odometry wheels should be mounted in line with the rear axle. Such a configuration minimizes errors due to slippage and avoids changes in the wheel's circumference due to variations in vehicle load.

Fig 3.2 shows the structure of a mobile robot and the arrangement of two driving wheels driven by two dc servo motors and the two measuring wheels that rotate independently of the driving wheels.

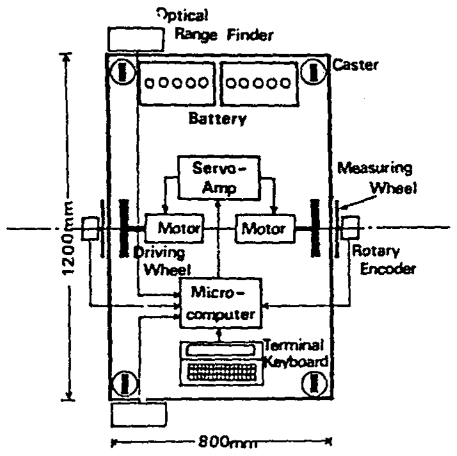
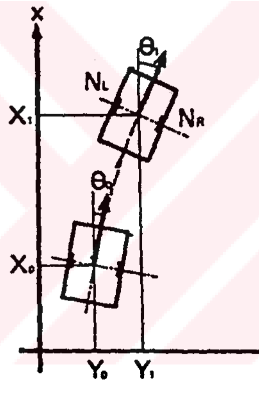


Figure 3.2. Schematic diagram of a mobile robot

The location unit is always calculating the current position and heading of the vehicle. The displacement of the vehicle for a short period is calculated from the rotational displacements of the measuring wheels, which are detected by rotary encoders. Then the current position and heading of the vehicle are obtained by adding its displacement to the last calculations [33]. The principle of this calculation is shown in Fig 3.3. The current position and heading are used to determine the current command and the rotational speeds of the driving wheels. Odometry requires only a single integration instead of the double integration of accelerometer-based devices. Consequently, the uncertainty grows only as a function of time rather than time squared.



$$\theta_1 = \theta_0 + \frac{N_L - N_R}{\text{Tread}}$$

$$X_1 = X_0 + \frac{N_L + N_R}{2} \cos \frac{\theta_0 + \theta_1}{2}$$

$$Y_1 = Y_0 + \frac{N_L + N_R}{2} \sin \frac{\theta_0 + \theta_1}{2}$$

Figure 3.3. Principle of position and heading measurement

3.2. Inertial Navigation Systems :

Inertial navigation system has been adapted from aerospace navigation. It uses a kind of dead reckoning in that position in two-or three-dimensions is calculated as the double integral of acceleration, measured by accelerometers on a gyrostabilized platform.

The primary sensor, in an inertial navigation system is the accelerometer. This instrument produces a precise output. In either analog or digital form, which is proportional to acceleration applied along the input axis of the sensor. It is necessary that the accelerometer assembly be referenced to a coordinate system which can be maintained or defined in a precise manner. The nature of this reference coordinate system depends on the nature of the vehicle and its mission. For example, a manned aircraft or a submarine uses an Earth referenced coordinate system, while a space vehicle is concerned with a inertial reference [34]. Regardless of coordinate system, it must be handled consistently and information can be provided as needed. To accomplish the orientation control of the accelerometers, the gyroscope which is an inertial sensor, is used. The gyroscope has the required characteristics of being able to prescribe a reference in inertial space. The gyroscopes and accelerometers are generally mounted in a cluster arrangement which is then gimballed or strapped down to measure vehicle motions about and along axes. Gyros and gimbals are used in conjunction with electronics and gimbal torquers to create null-seeking servo loops for the gimballed case. Any angular motion about the axes is sensed by the corresponding gyro and via appropriate gimbal control maintains the cluster fixed within the reference frame. Fig 3.4 displays a schematic of a typical two DOF gyro in its gimbals. It provides a measure of angular displacement of the case about the two orthogonal axes. Tilting of one gimbal axis produces torque, known as presession, at the perpendicular axis. Pickoffs at the gimbal pivot points provide a reference signal to indicate the gyro's angle in space.

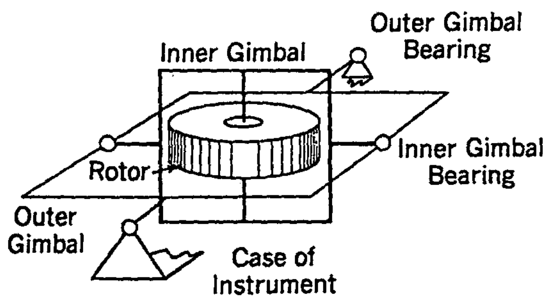


Figure 3.4. Schematic of Two Degrees-of-Freedom Gyro

Output transducers on the gimbals provide attitude output. Or the mechanical assembly of inertial sensors can be strapped down and computationally a reference attitude matrix is determined which is effectively the stabilized reference system. The two approaches give the same results.

While the mobile robot moves, the accelerometer sense the acceleration and when properly integrated, yields the velocity and distance traveled of the robot. As it is remarked inertial navigation system requires double integration consequently, the uncertainty grows as a function of time squared.

A typical gimballed system and a strapdown inertial system as well as their major component parts are shown in Fig 3.5.

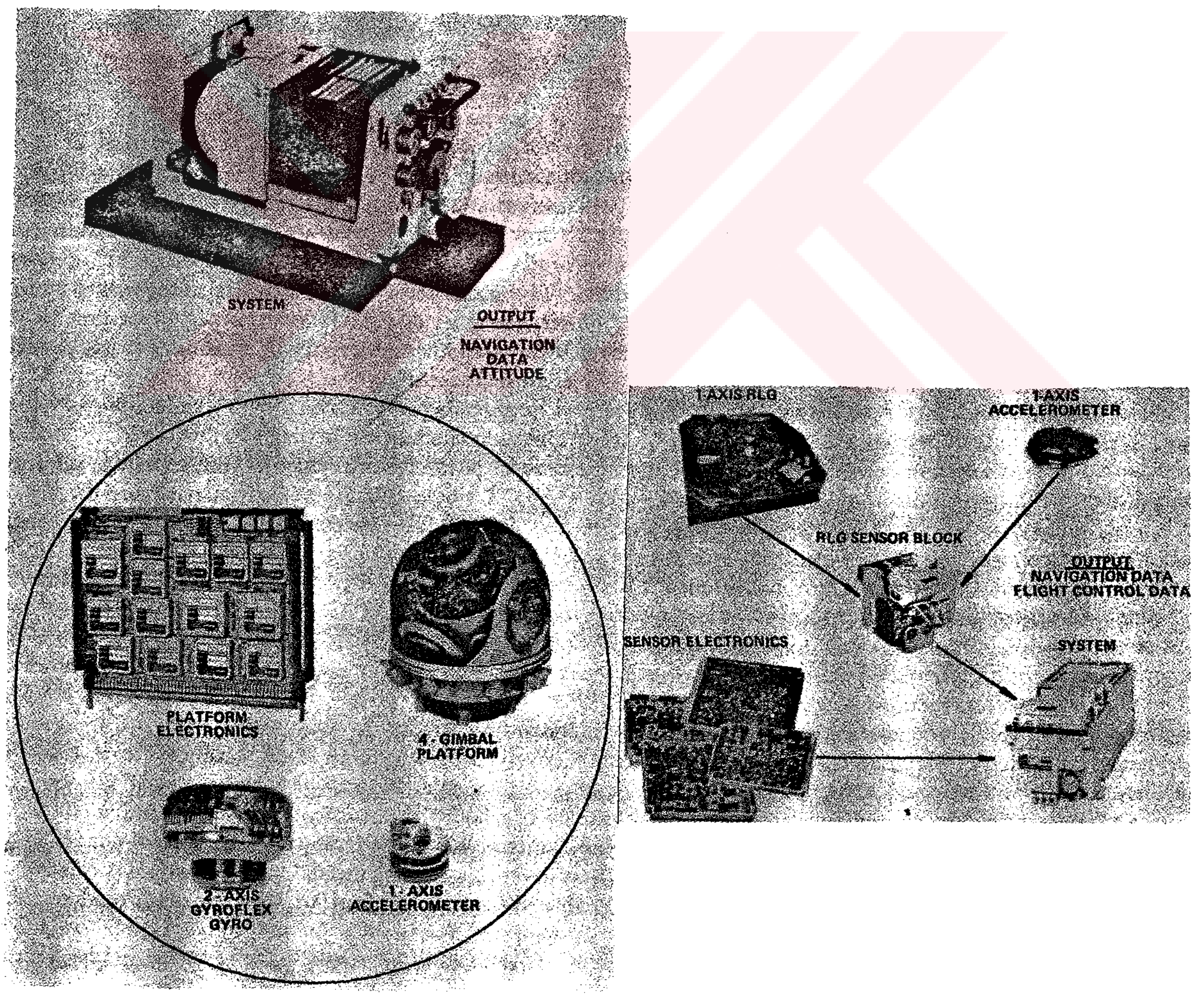


Figure 3.5 Inertial Systems

A wide array of components and installations for inertial navigation systems is available as presented in the following:

A. Gyro Types:

Single-degree-of-freedom, two-degree-of-freedom, free rotor and solid state including:

1) Rotating Wheel:

a) Wheel within float in buoyant fluid

1) Jewelled bearing support,

2) Magnetically supported,

3) Other exotic support.

b) Wheel supported by universal joint (hinge)

1) Torsional hinges,

2) Flex hinges.

c) Wheel or rotor electrostatically supported.

d) Momentum element support

1) Ball bearing,

2) Gas bearing,

3) Fluid bearing,

4) Electrostatic bearing.

2) Optical Gyro (Solid-State Gyro):

a) Ring Laser Gyro: The basic operation of the RLG is to excite and sustain two oppositely directed traveling waves that can oscillate with different magnitudes and frequencies. The frequency difference between the two traveling waves under ideal conditions is to direct function of the inertial rate perpendicular to the plane of traveling waves.

b) Fiber Optic Gyro: FOGs have the potential of being simpler, more reliable and less costly to build than RLGs. The most popular FOG is a Sagnac interferometer. The basic idea is that light traveling in clockwise and counterclockwise directions through a fiber-optic coil emerges with a slight time and hence optical phase difference between the two paths when the coil is rotating about the axis of the coil.

3) Nuclear Magnetic Resonant Gyro:

4) Multisensor-Combined Gyro and accelerometer.

B. Accelerometer Types:

These include single, dual axis, freely supported and the following:

1) Proofmass Supported by Pendulum

a) Electromagnetic restoring loop (analog or digital)

b) Proofmass supported by beam or string or turning fork in oscillator configuration

c) Proofmass supported by gyroscope

d) Proofmass supported electrostatically with appropriate readout capability.

C. Reference Platform Mechanization Approaches:

a) Four gimbals utilized to support a cluster of gyros and accelerometers.

b) Three gimbals similarly used with attendant loss of gimbal freedom.

c) Strapdown-the computer utilizes gyroscopic outputs to computationally establish the reference desired coordinate frame.

d) Hybrid strapdown-gimballed; a combination of gimballed electro-servo isolation and strapdown reference action.

e) Multifunction platform assembly- a combination of inertial and aided sensor operation

In Navlab autonomous vehicle [35], an inertial navigation system to acquire the position and velocity of the vehicle has been used. It consists of three ring laser gyros, which measure angular acceleration about three orthogonal axis, and three accelerometers, which measure linear acceleration in three inertial and orthogonal directions.

The actual design of the inertial system obviously depends upon accuracy requirements, mission environments, usage time, reaction time, reliability, cost and many other factors. These sensors although currently expensive one axis gas rate gyros cost approximately $ 2000 in 1989- are expected to decline in price, while performance is expected to improve steadily, especially for ring laser gyros. If such projections materialize, it can be expected conventional inertial guidance sensors to be increasingly used.

3.3. Vision and Direct Imaging Systems:

Vision and direct imaging systems are particularly appropriate sensors to use in mobile robot navigation applications. Range information plays an important role because from it, information can be easily derived concerning the mobile robot's position, areas of free space and the presence of obstacles.

Stereo ranging, as it is shown in Fig. 3.6, determines the range to an object when two images are available, either through two cameras, or through the movement of a single camera .

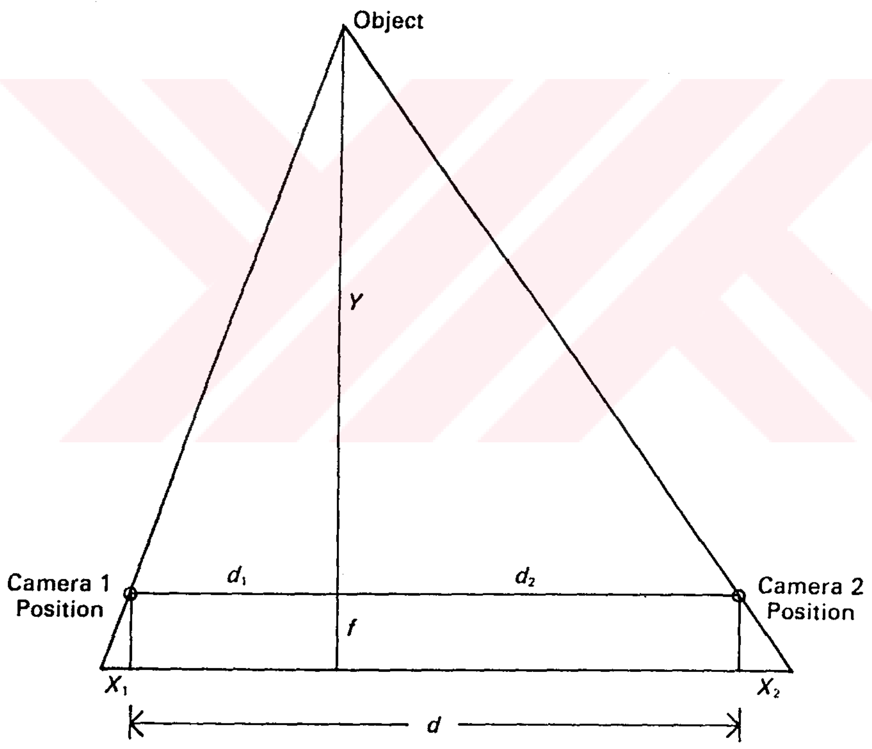


Figure 3.6. Stereo ranging

In this approach, it is assumed that the two sensor positions are located in a plane parallel to the image plane. By similar triangles the range Y can be obtained.
Since,

$$d_1 = d - d_2, \qquad (3.10)$$

$$\frac{Y+f}{f}=\frac{d-d_2+x_1}{x_1}=\frac{d_2+x_2}{x_2} \qquad (3.11)$$

Rearranging and simplifying the last two terms gives,

$$\frac{d}{x_1}=\frac{d_2}{x_2}+\frac{d_2}{x_1} \qquad (3.12)$$

solving for d_2, it is obtained.

$$d_2=\frac{dx_2}{x_1+x_2} \qquad (3.13)$$

Substituting in equation (3.11) and simplifying then gives,

$$Y=\frac{df}{x_1+x_2} \qquad (3.14)$$

In this formula, x_1 and x_2, corresponding pixel positions of the left and right camera respectively, are variables that determine the object range, and f and d are held as constants.

Depth accuracy is limited by the number of pixels available and nonlinearities in the image. Accuracy is not good for objects not located between the two cameras. finding the same point in each picture can be quite difficult. Known as the pixel correspondence problem, there are two basic approaches to solving it. One method searches for a characteristic pattern, the other uses correlation techniques within a moving window.

As a stereo ranging system application, The Stanford Cart which is a remotely controlled TV equipped mobile robot at the Stanford Artificial Intelligence Lab. can be given [36]. The Cart gains its knowledge of the world entirely from images broadcast by the on-board TV system. The Cart uses several kinds of stereo vision to locate objects around it in 3D and to deduce its own motion. The system is reliable for short runs, but very slow. A run began with a calibration of the Cart's camera. The Cart was parked in a standard position in front of a wall of carefully painted spots. The Cart's camera's focal length and distortions were determined by parking the Cart a precise distance in front of, and aiming its camera at, a carefully painted array of spots pattern and running a calibration program. The program located the spots in the image and fitted a 2D, third-degree polynomial which converted actual positions in the image to coordinates in an ideal unity focal length camera. The cart moves 1m every 10 to 15 minutes, in lurches. After rolling a meter it stops, takes some pictures and thinks about them for a long time then it plans a new path and executes a little of it and pauses again. At each pause on its computer-controlled itenerary, the Cart slid its camera from left to right on a 52-cm track, taking nine pictures at precise 6.5cm intervals. Points were chosen in the fifth of these nine pictures. The camera slid parallel to the horizontal axis of the camera coordinate system, so the parallax-induced displacement of features in the nine pictures was purely horizontal. The Correlator is a subroutine that, given a feature description produced from one image, found the best match in a different, but similar, image. Its search area could be the entire new picture, or a rectangular subwindow. The Correlator was applied eight times to look for the points chosen in the central image in each of other eight pictures. The search was restricted to a narrow horizontal band. This had little effect on the computation time, but it reduced the probability of incorrect matches. In the case of correct matches, the distance to the feature was inversely proportional to its displacement from one image to another. The uncertainty in such a measurement is the difference in distance a shift one pixel in the image would make. The uncertainty varies inversely with the physical separation of the camera positions where the pictures were taken. Long stereo baselines give more accurate distance measurements. After the correlation step, the robot knew a feature's position in nine images. It considered each of the 36 possible image pairings as a stereo baseline and recorded the inverse distance of the feature in a histogram. Each measurement added a little normal curve to the histogram, with mean at the estimated distance, and the standard deviation inversely proportional to the baseline, reflecting the uncertainty. The area under each curve was made proportional to the product of the goodness of the matches in the two images, reflecting the confidence that the correlations were correct. The distance to the feature was indicated by the largest peak in the resulting histogram as it is shown in Fig 3.7, if this peak was above a certain threshold. If below, the feature was forgotten.

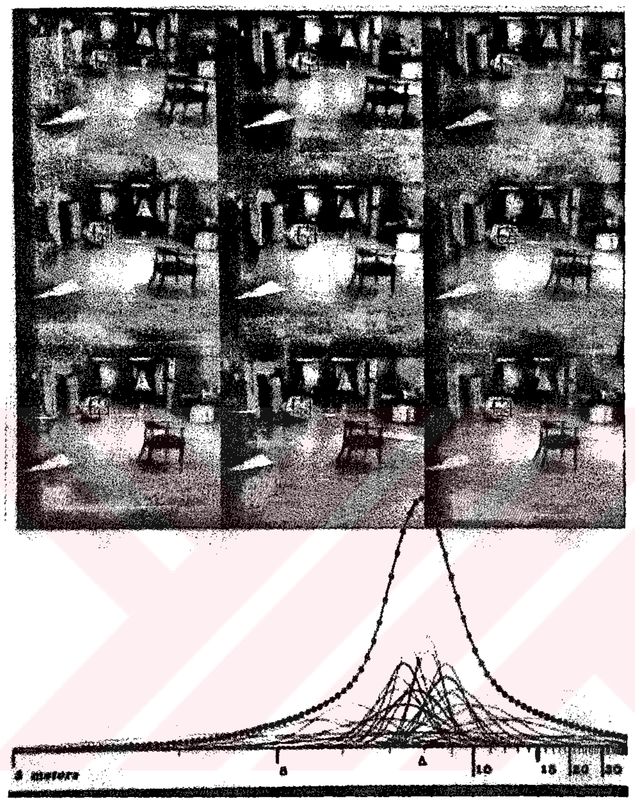


Figure 3.7 A typical stereo ranging

After having determined the 3D location of the objects at one position, the computer drove the Cart about a meter forward. At the new position, it slid camera and took nine pictures. The Correlator was applied in an attempt to find all the features successfully located at the previous position. Feature description extracted from the central image at the last position were searched for in the central image at the new stopping place. Slider Stereo than determined the distance of the features so found from the Cart's new position. The program now knew the 3D position of the features relative to its camera at the old and the new locations. Its own movement was deduced from 3D coordinate transform that related the two.

Direct imaging systems, namely, optical and laser range finders are also used as mobile robot navigation systems. A simple low-cost optical range-finder has been developed specifically for cart navigation [37]. A 2mW 0.82μ LED source, which is not a laser, is 100% amplitude modulated at 5Mhz and used to form a collimated 1-in-diameter transmit beam that is unconditionally eye-safe. Returning scattered radiation is focused by a 10.16cm diameter coaxial Fresnel lens onto a p-i-n silicon photodiode. Range is determined from the phase shift between the 5Mhz modulation on the transmitted and received signals. Use of rotating mirror provides 360 degrees polar coordinate coverage of both distance and reflectance out to about 6.1m around the vehicle. Both radial and angular resolution correspond to about 2.54cm at a ranging distance 1.524m, with overall bandwidht of approximately 1kHz. The system is capable of reading wall bar codes on the fly and in addition capable of simultaneously ranging and acting as a wideband optical communication receiver.

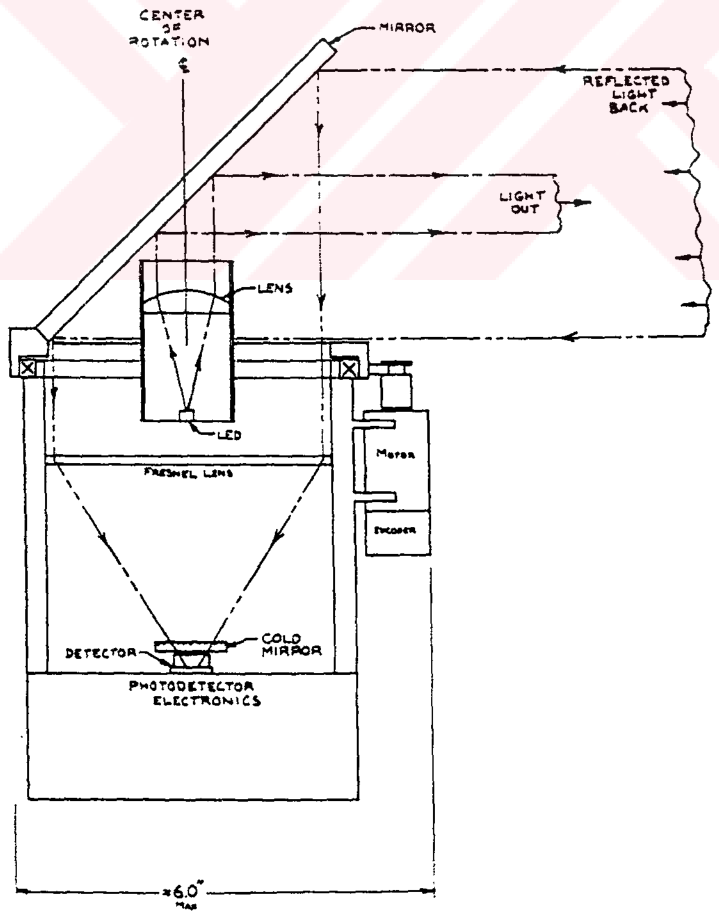


Figure 3.8 Pyhsical construction of the optical ranger

Total parts cost for the optical transmitter, Fresnel lens, receiver and all the electronics is less then $200. The remaining major parts, consisting of the rotating mirror, ring mounting, motor and incremental encoder, costs less then $500. In Fig 3.8, the physical construction of the optical ranger can be seen. The receiver electronics are mounted on one 10.16cmx10.16cm board directly below the photodetector. The optical ranger has been mounted on an research tricycle cart, named Blanche, having one steered and powered wheel and two trailing idling wheels. Initially cart navigation was by odometry augmented by the use of a gas-jet gyroscope for additional rate-of-turn information [15]. The added capability of the optical ranger, mounted above the cart, as shown in Fig. 3.9, allows the vehicle to sense its surroundings. The ranger is mounted directly above the cart in such a way that it can either provide a radial scan or, by tilting the mirror, a conical scan to detect any object in its path. An alternative mounting is in the space between the bottom of the cart and the floor.

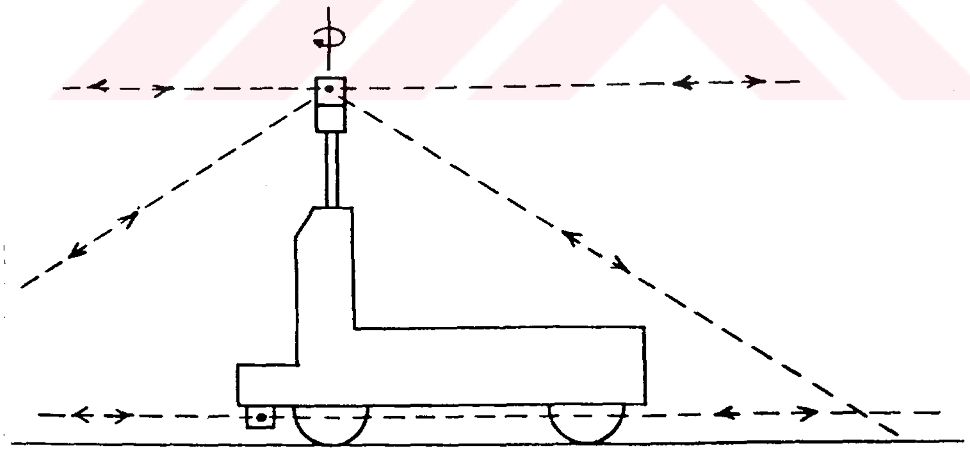


Figure 3.9 The optical ranger mounted above the cart

A typical sample of the ranger output is shown in Fig 3.10.

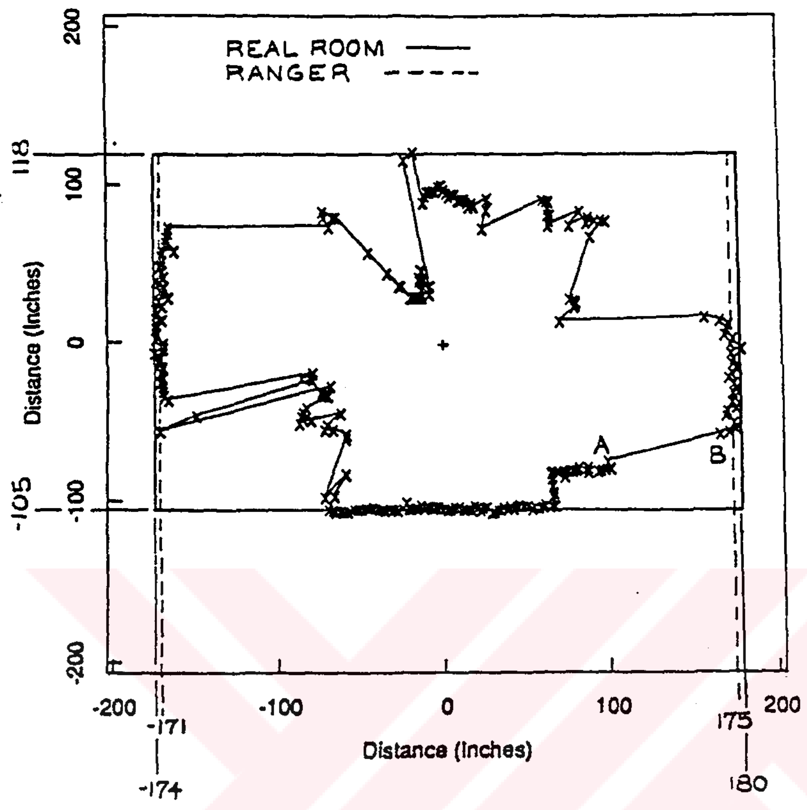


Figure 3.10 A representative ranger scan

The ranger is located at the (0,0) position, marked by cross, in the center of the plot. The data collection correspond to a single mirror rotation taking about 1 second. The display computer program employed connects adjacent data points with straight line segments. The reason for the absence of any data points in regions such as that labeled AB in Fig 3.11, is that in such ranges the signal is outside the amplifier gain monitor and therefore no data are accumulated. The expected operating environment will usually be an aisle in a factory, having a total width of about 2m to 3m and essentially indefinite length. The vehicle will therefore typically be continuously interacting with a roughly linear length of artifacts perhaps about 90cm to 150cm or so cm away on either side, and observed for maybe up to approximately 25cm ahead and behind on either side. The ranger was designed with this environment in mind. as the vehicle moves forward, the data in the slice of the world seen by the ranger flows by on either side. New features are continuously added at the front while old ones drop out at the back. Most of the data between these two vision extremes is of course both highly correlated and redundant from view-to-view since it is essentially the same information continuously translated in space and seen from a slowly changing angular viewpoint. Also the odometry information is available to continuously cross check and augment that from the ranger and vice versa. Therefore odometry and optical ranging will be able to provide reliable indoor cart navigation when suitably combined with stored map information and appropriate heuristics.

Unlike other active range sensors, laser range sensors are more precise, less sensitive to lighting conditions and scan over much longer distances. Their still high cost is their biggest disadvantage . For example the Odetics laser range-finder, capable of scanning 128x128 field of view in 835 ms and ranging to 9.14m with a resolution of 1.778cm cost in the region of $125.000 in 1989 [38]. Fig 3.11 shows a schematic representation of a laser range scanner called "The Cyclone".

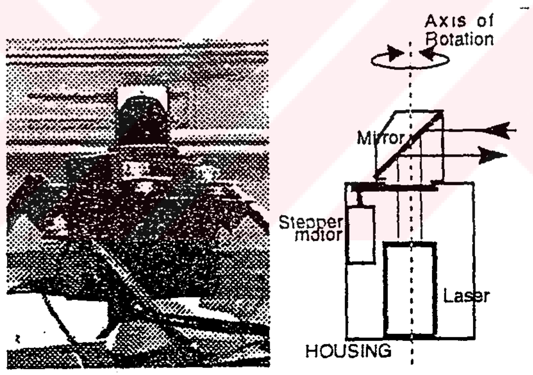


Figure 3.11 The Cyclone laser range scanner

It acquires fast, precise scans of range data over long distances(up to 50m). The components of the Cyclone are a pulsed gallium arsenide infrared laser range sensor, a rotating mirror, a stepper motor, and an encoder mounted on a hard steel housing. The Cyclone returns distance as a measure of the amount of time it takes for a transmitter beam to bounce off a target and be detected by a receiver. The laser beam projected by the transmitter is deflected horizontally by the mirror. By rotating the mirror, the Cyclone is able to scan a plane perpendicular to the axis of rotation and construct a two-dimensional map of the environment of 360 degrees field of

view. The resolution of the range measurements is set to be 10cm and the accuracy is ±20cm.
The angular resolution depends upon the resolution of the encoder that is used on the tower motor. The scanning speed is programmable between 0.5 and 5 revolutions per second.

The measurements of range-finders are not free of error, namely:

1. The range measurements are truncated to the resolution of the sensor.

2. Deviation of the range measurement from the exact distance due to noise and other factors inherent to the sensor.

3. Spurious, caused by specular reflections, small landing angle and mixed point effect at the edges.

4. Errors due to misalignment in the geometrical configuration of the sensor. Since the sensor is expected to scan in a plane parallel to the ground, a tilt in the laser or optical emitter, in the mirror or in the housing will affect to the scan surface.

5. Deviation of the line-of-sight from the nominal angular value caused by small errors in angular positioning of the rotating mirror. Its magnitude is proportional to the range measurement.

To achieve goal-directed autonomous behavior, the vision system for a mobile robot must locate and model the relevant aspects of the world . For an outdoor autonomous vehicle, it is necessary to find and follow roads using a camera. Two methods could be used: line tracking and region analysis. The first approach attempts to extract the edges of the road as seen by a camera, and backprojects them on the ground plane in order to compute the appropriate steering commands for the vehicle. This method assumes an accurate model of the road shape. Unfortunately, in practice the strongest edges are often the shadow edges, whereas the road edges are much weaker and do not necessarily fit the straight line model. The line tracking approach relies heavily on inferring the road geometry from the road edges. The geometric inference assumes that the road edges are parallel and therefore it is very unstable when applied to noisy data. The second method classifies the pixels of a color image as on-road or off-road pixels based on the color characteristics of the road. This method does not require a geometric model as accurate as the line tracking approach. However the main problem is that the dominant color features are dark(shadows) and light(sun). As a result, road and non-road pixels cannot be separated by a single threshold.

The region analysis method has been used in Navlab 39]. Navlab is based on a commercial van chassis, with hydraulic drive and electric steering. It is currently equipped with two sensors mounted above the cab: a TV camera, and a laser range finder. The road following algorithm runs in about 10 s per image on a dedicated Sun 3/160, using 480x512 pixels images reduced to 30 rows by 32 columns. A new image is processed every 4 m, which gives about three fourths of an image overlap between images. The road following algorithm which has been used in Navlab involves three stages:

Step 1. Classify each pixel:

Pixel classification is done by a standard pattern classification method. Each class i is represented by the means m_i and covariance matrix Σ_i, of red, green, and blue values, and by its a priori likelihood, f_i, based on expected fraction of pixels in that class. Assuming Gaussian distribution, the confidence that a pixel of color X belongs to a class is computed based on the distance. Each pixel is classified with the class of highest probability.

Step 2. Use the results of classification to vote for the best-fit road position:

Once each point has been classified, the most likely location of the road must be found. The road is assumed locally flat, straight and has parallel sides. The road geometry can be described by two parameters as shown in Fig. 3.12.

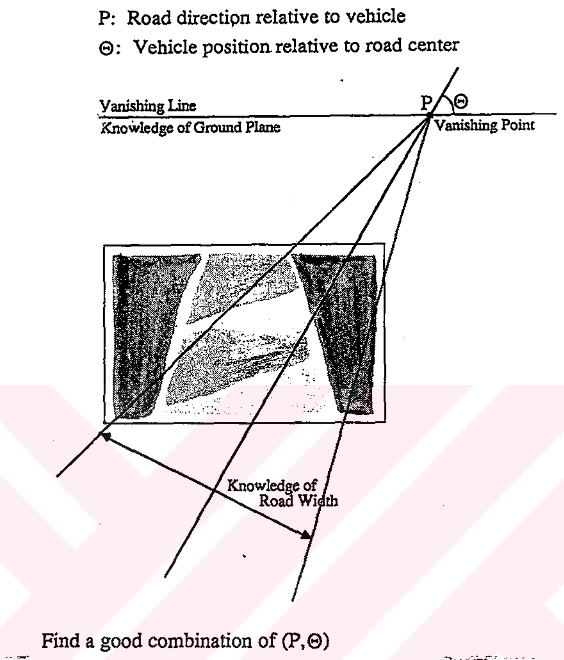

P: Road direction relative to vehicle
Θ: Vehicle position relative to road center
Vanishing Line
Knowledge of Ground Plane
P
Θ
Vanishing Point
Knowledge of Road Width
Find a good combination of (P,Θ)

Figure 3.12. The parameters of the road geometry

The intercept P, which is the image column of the road's vanishing point. This is where the road centerline intercepts the vanishing line of the locally flat plane of the road. In other words, the intercept gives the road's direction relative to the vehicle. Since the camera is fixed to the vehicle this vanishing line is constant assuming that the vehicle's pitch and roll relative to the road are small. The orientation θ of the road in the image, which tells how far the vehicle is to the right or left of the centerline. A two- dimensional parameter space, with intercept as one dimension and orientation as the other. Each point classified as road votes for all road (P, θ) combinations to which it could belong, while nonroad points cast negative votes. For a given pixel (row,column), the corresponding set of pairs (P, θ) is the curve:

$$(\text{column} + (\text{row} - r_{\text{horizon}}) \times \text{tg}\theta,\ \theta) \qquad (3.15)$$

in parameter space. The (P, θ) pair that receives the most votes is the one that contains the most road points, and it is reported as the road.

Since this method does not rely on the exact local geometry of the road, it is very robust. The road may actually curve or not have parallel edges, or the segmentation may not be completely correct, but still the method outputs approximate road position and orientation. The main limitation is the field of view of the camera which limits the size of the parameter space. In order to convert the image position into vehicle coordinates, the system is first calibrated by computing for two rows r_1 and r_2 the number of pixels per meters, ppm_i and the column corresponding to the center line of the road c_i. The calibration assumes that the widht of the road w is known, and the distance d_i between the vehicle and each row r_i is known. The location in the vehicle coordinates of a road of image coordinates (P, θ), is given by the vehicle coordinates of the intersections Q_i, between the line of parameters (P, θ) and the rows r_i as it is shown in Fig 3.13.

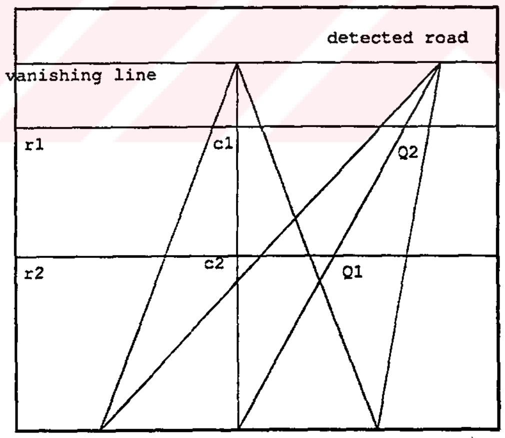


Figure 3.13 Calibration procedure

If Q_i is at column C_i in the image, then the distance between Q_i and the center of the vehicle is given:

$$x_i=(C_i-c_i) / ppm_i \qquad (3.16)$$

Step 3. Collect new color statistics based on the detected road and nonroad regions.

Once the road has been found in an image, the color statistics of the road and off-road models are modified for each class by resampling the detected regions and updating the color models. The updated color statistics will gradually change as the vehicle moves into a different road color, as lighting conditions change, or as the colors of surrounding grass, dirt, and trees vary. As long as the processing time per image is low enough to provide a large overlap between images, the statistics adapt as the vehicle moves. The road is picked out by hand in the first image . Thereafter, the process is automatic, using the segmentation from each image to calculate color statistics for the next. The occasional remaining problems in region analysis technique in road following come from one of three causes:

- The road is covered with leaves or snow, so one road color class and one nonroad color class are distinguishable.
- Drastic changes in illumination occur between successive pictures so all colors change dramatically from one image to the next.
- The sunlight is so bright and shadows are so dark in the same scene that the hardware limits of the camera are hit. It is possible to have pixels so bright that all color is washed out, and other pixels in the same image so dark that all color is lost in the noise.

Another autonomous outdoor vehicle which the system configuration is shown in Fig. 3.14 is ALV (Autonomous Land Vehicle).

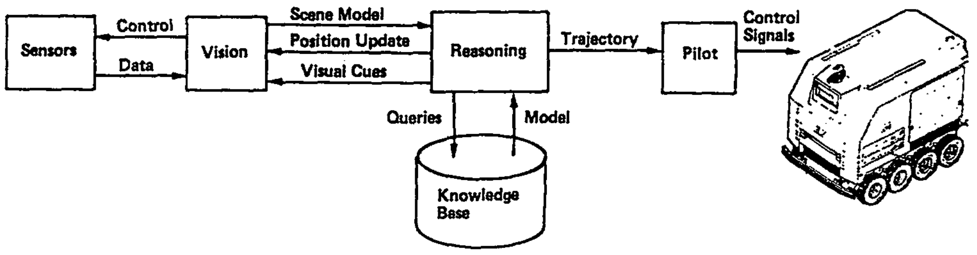


Figure 3.14 The system configuration of ALV

Two imaging sensors are available on the ALV [40]. The primary vision sensor is an RCA color video CCD camera, which provides 480x512 red, green and blue images, with eigth bits of intensity per image. The field of view (38 deg. vertical and 50 deg. horizontal) and focus of the camera are kept fixed. The camera is mounted on a pan/tilt unit that is under direct control of the vision subsystem. The other vision sensor is a laser range scanner. This sensor determines range by measuring the phase shift of a reflected modulated laser beam. The laser is continuously scanned over a field of view that is 30 deg. vertical and 80 deg. horizontal. The output of the scanner is a digital image consisting of a 64x256 array of pixels with 8 bits of range resolution. Vision system of ALV is used for range-based road following. The responsibility of the vision system in road following is to process data in the form of video or range images to produce a description of the road in front of the vehicle. This description is passed to the reasoning subsystem of ALV, which is additional data such as current position, speed and heading to generate a trajectory for ALV to follow. The road is presented by its edges. The video data processing unit uses a clustering algorithm to segment the image into road and nonroad regions. After producing a binary road image, the road boundaries are traced and selected image points are transformed into three-dimensional road boundary points. The algorithm for video-based road following as follows:

1. Digitize the video images

2. Segment road and nonroad regions

3. Extract road boundaries by tracing the binary road edges

4. Transform 2D edge points to 3D coordinates and build the scene model

Fig. 3.15 shows the vision system flow of control in a complete scene model cycle.

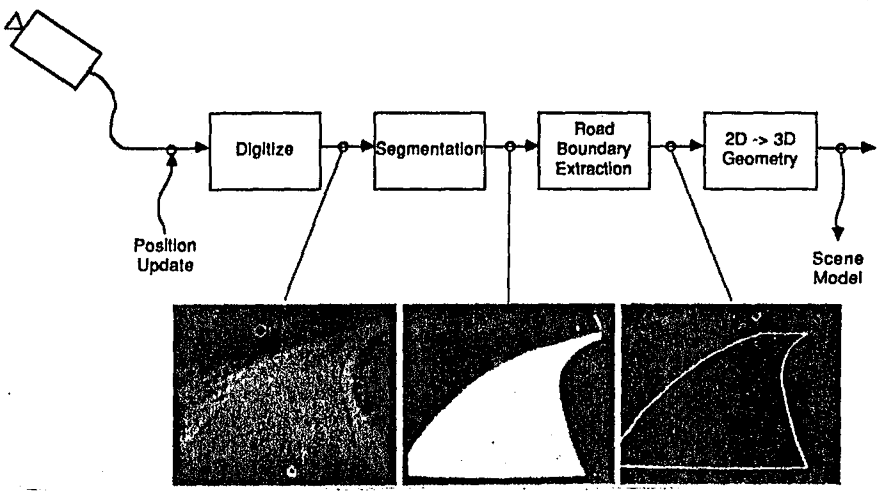


Figure 3.15. Vision system flow of control

For mobile robots, the design of a real-time vision system plays an important role in navigation. There are basically two approaches to the design of a real-time vision system. One is using extremely fast hardware elements and possibly a massively parallel structure, yielding a supercomputer with an impressive power in terms of the notorious million instruction per second. The other one is to look for the inherent structure and, possibly and to find a computer architecture which is well matched to the task of visual motion control. The second approach called dynamic vision, indeed, feasible and has led to the construction of a family of multiprocessor systems specialized for it [41]. The architecture of these real-time systems is very different from that of a typical image processing system. In spite of their relative simplicity, they have proven to be a very powerful hardware basis for various real-world experiments where mechanical systems or vehicles were controlled by dynamic vision. An important concept upon which to base the design of a dynamic vision system is temporal continuity. Usually natural scenes change only gradually, and if two pictures of such a scene are taken within a few milliseconds, they will normally be a very similar to each other. In order to understand how the temporal continuity of natural scenes can facilitate dynamic vision, assume that a first TV image of such a scene has just been interpreted. It is than rather easy to interpret the immediately following image, as the differences between the two are very small. This observation has important consequences for the design of a real-time vision system. It means that the task of dynamic scene interpretation becomes easier if the time spend on each image is reduced, and the task becomes more difficult if the system is slower. Therefore, the cycle time of the low-level vision subsystem should ideally be less than one frame period of the TV signal used, making it possible to evaluate every simple image as it delivered by the camera. The appropriate behavior of a vision controlled machine typically depends on the presence and location, or absence of a certain objects in its environment. The vision task is then clearly goal directed, the first subtask is being to locate features in the image which are indicative of the presence and location of important objects. It seems obvious that such features in many typical situations occupy only a small fraction of the total area of each image as it is shown in Fig 3.16.

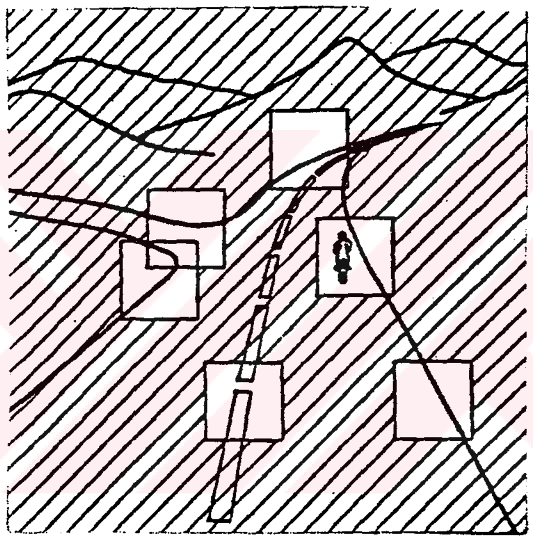


Figure 3.16. Small regions of an image

It suffices then to process only those areas of each image which actually contain relevant features. In dynamic scene interpretation, the location of all important features is usually known in advance and with fairly good precision from the interpretation of previous images. This means that when interpreting the next image in the sequence, the search space in which the feature of interest should be looked for is small, and the feature can be rediscovered rather quickly if the search is indeed focused on this small search space. This leads to the probably most important point in the design of hardware for real-time vision since nearly all the relevant information in the image is contained in a limited number of small regions, the combined size of which is only a small fraction of the whole image, much will be gained if all the available computing power can be concentrated on these regions. Moreover, since each region may contain a different type of feature, it is important to be able to use the different algorithms in each region. This shows that a conventional image processing system which is designed to treat all pixels in an image in the same way does not have the proper structure for dynamic vision because they must treat all pixels of an image which are known in advance to contain no relevant information. In the worst case, additional computing power is needed to delete all the irrevelant data which are produced in the process. The concept of processing only a limited number of well defined regions within an image is also the key to a natural division of the problem into subtasks which can be executed in parallel or a coarsly grained multiprocessor system. Each parallel processor in such a system can be assigned one relevant region and it can locate independently of all other processors-the associated features in that region. Such a system not only has a very clear structure, but it can also be very efficient, since the parallel processors do not have to spend time synchronizing or coordinating each other. An important key to this concept is that the size, shape and location of each region may be varied during the interpretation process in a data dependent way. Each region will normally be continuously adjusted in such a way as to completely contain a relevant feature or object. If the regions were fixed, the system would be much less efficient for two reasons: First, some regions would contain no relevant information but would nevertheless absorb computing power; secondly, some features or objects would be dissected by the borders between regions, creating the difficulty of detecting and interpreting arbitrarily dissected parts, representing them internally and finally recombining them into objects. The speed is a most critical characteristics of any feature extraction. In static image processing very little is often known in advance of the image presented to the system. The task then is to extract as much information from the image as possible. This is very different from dynamic vision, where each new image is known to be a natural continuation of a sequence of images which the system has interpreted already, differences relative to the previous image to be expected in small details only. Most of the time, the feature extraction has to answer only a small number of precise questions relating to one or another of the small differences, such as how much and in which direction did a certain feature move in a small fraction of a second. As an example of a first dynamic vision system, BVV1, a multiprocessor system can be given [42]. It could contain up to 15 microcomputers: One system processor (SP) and up to 14 parallel processors (PP) as it is seen from Fig 3.17.

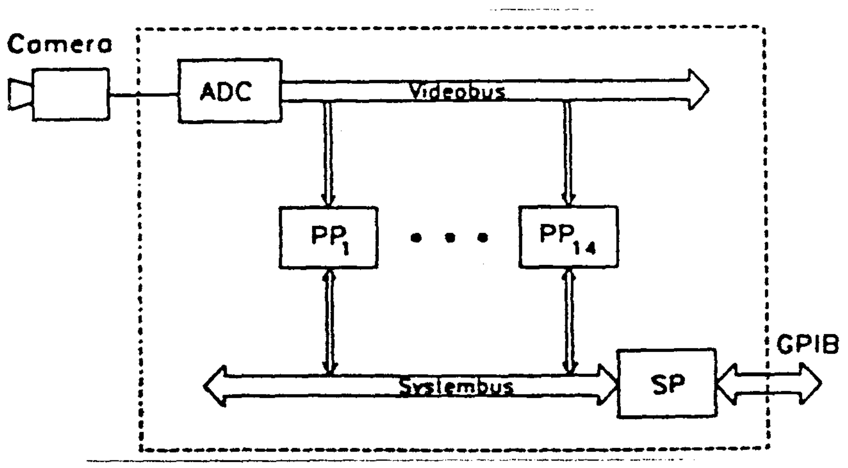


Figure 3.17 The BVV 1, a multiprocessor system for dynamic vision

All processors use the Intel 8-bit microprocessor 8085 as a CPU and have 16kByte of memory. The video signal is digitized by an analog to digital converter (ADC) and distributed in digital form via the videobus to all PPs. Each PP copies that part of the image which constitutes its momentary region of interest into an internal private memory. Having a private image memory in each PP avoids delays which would exist if a central image memory would be shared by all PPs. Each PP is a self contained computer executing its own task. Messages can be sent from any PP to any other PPs, or to an external host computer. The SP transports all messages, relieving the PPs completely from this task and coordinates all internal and external communication. General purpose interface bus GPIB connects the BVV1 to other equipment such as host computer.

According to the knowledge above, vision and direct imaging systems provide information that is difficult to obtain in other ways. Although a few vision systems have been available for about ten years, they have not been used because of their limited capabilities and high cost. However vision systems as mobile robot navigation systems are now used more and more frequently because of the development in real-time vision systems and decreasing prices. The main disadvantage of vision and direct imaging systems is that they require complex and time consuming analysis and computations which restrict the velocity of the mobile robot.

3.4. Ultrasonic Systems:

Ultrasonic systems are based on the transmission and reception of sound at frequencies above the range of human hearing. It has been long recognized that ultrasonic systems may be used to determine range and direction to a target. This technique is the basis of sonar. In this area, nature is ahead of the researchers, since bats have been employing ultrasonic principles for target tracking, navigation and collision detection. Fig. 3.18 shows the basic principle behind ultrasonic ranging.

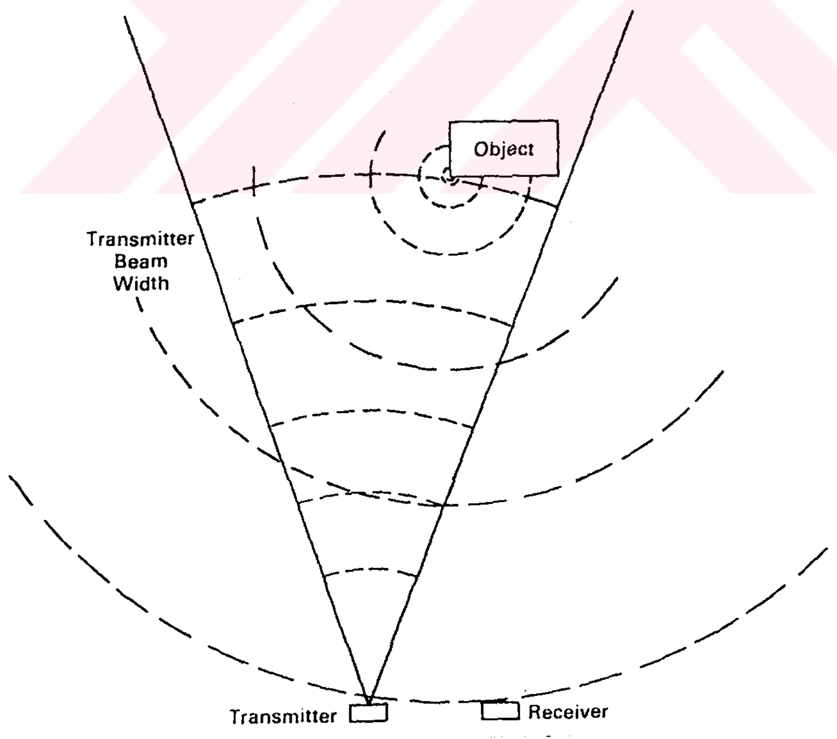


Figure 3.18 Basic sonar principle

When a sound transmission source sends out a pulse of acoustical energy, the resulting acoustical wave travels along a path defined by the beam width of the transmitting element. A portion of this energy is subsequently reflected back from any object that the sound waves reach. This returned energy can be detected by a receiving element, and the delay in time between the transmitted pulse and the received pulse can be measured. This delay time is in a direct proportion to the range of the object and allow the exact range to be calculated. The best frequencies for the most ultrasonic ranging systems are 40 to 80kHz.

Ultrasonic sensor is a device used to convert an electrical signal into sound energy and vice versa and is a key element in any ultrasonic system. Three types of ultrasonic sensors are available: Piezoelectric, electrostatic, and magnetostrictive. Compared to vision systems, ultrasonic systems are less expensive, faster, but they have some shortcomings. Their poor directionality limits the accuracy in determining spatial position of an edge to 10-50cm, depending on the distance to the obstacle and the angle between the obstacle surface and the acoustic axis. Frequent misreadings are caused by either ultrasonic noise from external sources or stray reflections from neighboring sensor. Specular reflections occur when the angle between the wavefront and the normal to a smooth surface is too large. In this case, the surface reflects the incoming ultrasound waves away from the sensor.

With careful design, ultrasonic ranging is possible with accuricies of 0.254cm in at 3m (<%0.1). With array processing and other techniques, azimuth resolutions as small as 1° or less are possible. One suggested approach [43] to gain equivalent angular accuracies as low as 1° and to improve azimuth resolution is to use the difference in arrival time of two ultrasonic returns to detect more accurately the angle to the target .

Fig. 3.19 shows a mobile robot that must travel to a goal point A. The robot has one transmitting and two receiving sensors.

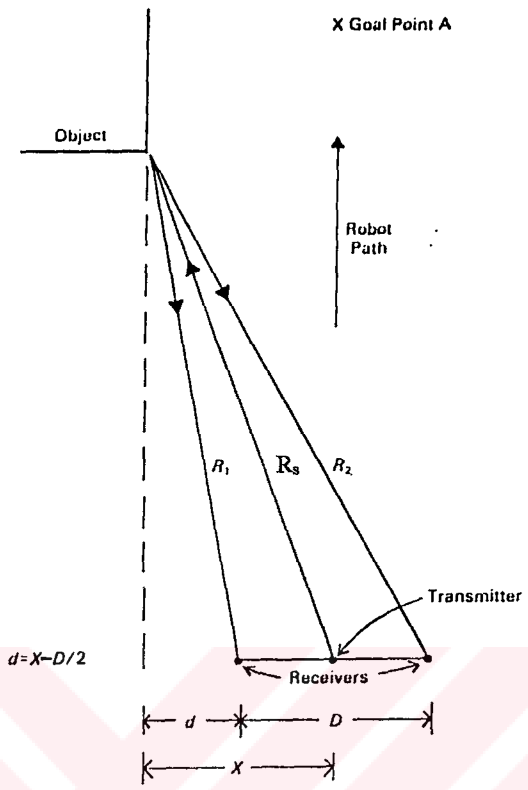


Figure 3.19 Difference of arrival time

To make sure, it will clear the obstacle shown, the robot must determine the horizontal distance to the object X and compare it to half of its own width. Horizontal distance may be determined as follows. Two receivers are located a distance D in apart at the front of the robot. A sonar transmitter is mounted between them in the center. Sonar signals emitted from the transmitter will be reflected from the corner of the object. Separate signals are receive by each sonar receiver. Knowing the transmission time and the time of each reception, the robot can accurately determine separate ranges (R_1,R_2) from each receiver to the object. With R_1,R_2 and D known, the slant range R_s to the object and the horizontal distance X can be computed.

$$R_s \approx \frac{R_1+R_2}{2} \qquad (3.17)$$

$$X \approx \frac{R_2^2}{2D} - \frac{R_1^2}{2D} \qquad (3.18)$$

There are some trade-offs in the design of any sonar system. The higher the selected frequency, the better the range resolution. The narrower the beam width, and the less likely it is to miss small targets, however, the lower the frequency, the greater the range in the air and the larger the system field of view. another trade-off is in the number of pulses transmitted. The more ultrasonic pulses that are transmitted, the greater the likelihood of detecting a target return, but the resulting range resolution is poorer and system minimum range is increased.

Ultrasonic systems can also be used to generate a map through a scan of the area because mobile robots need a map of their environment. A map needs to have a high resolution in order to obtain this, wide angle range measurements must be taken. If the sonar transducer is moved repeatedly over small steps, the resultant information can be used to plot the object locations in the environment of the mobile robot. Another mapping approach is to use multiple sensors around the robot to let it see in all directions at once, and then have the robot move around to cover the area from many view point.

In the system of Neptune mobile robot, there is no a priori map of its surroundings [44]. Instead, it acquires data from the real world through a set of sonar sensors and uses the interpreted data to build a sonar map of the robot's operating environment.

The sonar devices being used are Polaroid laboratory grade ultrasonic range transducers. These sensors have a useful measuring range of 0.27-10.67m. The beamwidth w at -3dB is approximately 15^0. The range accuracy of the sensors is on the order of ± 3.48cm. The control circuitry provided with the unit, which is designed to return the distance to the nearest sound reflector in its field of view is used. The sensor interface does not provide information on multiple echoes nor on the phase shift or the intensity of the detected echo. The sonar sensor array consists of a ring of 24 Polaroid ultrasonic transducers, spaced 15^0 apart. Controlling microprocessor selects and fires sensors, times the returns and provides the corresponding range value. In this implementation, range measurements are obtained from sonar units whose position with respect to the robot is known. Each sonar range measurement is interpreted as providing information about probably empty and somewhere occupied volumes in the space subtended by the sonar beam, which is a 30° cone in front of the sensor. Fig. 3.20 shows a two dimensional projection of the conical field of view of an ultrasonic sensor. A range reading d indicates the existence of an object somewhere within the region A.

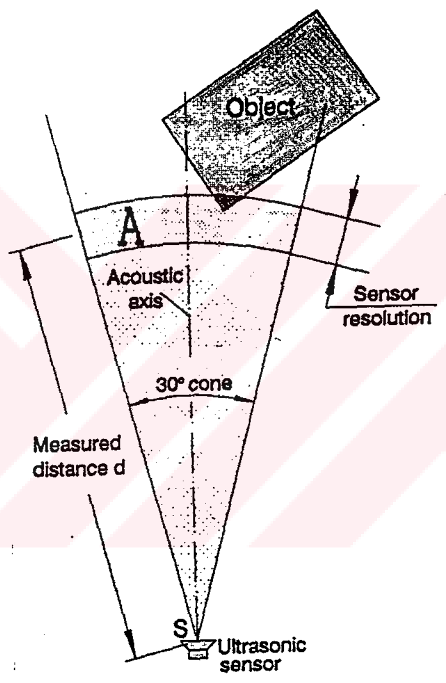


Figure 3.20 Range-finding using ultrasonic sensor

A number of problems that are inherent to the data obtained from the sensor device and intrinsic to the sensor itself:

- The timing circuitry limits the range precision.
- The detection sensitivity of the sensor varies with the angle of the reflecting object to the main beam axis.
- Sonar beams can suffer multiple reflections or specular reflections away from the sensor, giving false distance readings

Because of the relatively wide angle of the sonar beam, an isolated sonar reading imposes only a loose constraint on the position of the detected object. These problems preclude a direct interpretation of the sonar readings and a probabilistic approach to the interpretation of range data should be considered. This occupancy information is modeled by probability profiles that are projected onto a rasterized two-dimensional horizontal map, where empty, occupied and unknown areas are represented. Sets of range measurements taken from multiple sensors on the robot and from different positions occupied by the robot as it travels provide multiple views that are systematically integrated into the sonar map. In this way, the accuracy and extent of the sonar map are incrementally improved and the uncertainty in the positions of objects is reduced. Overlapping empty areas reinforce each other, the same happenning with occupied areas. Additionally, empty regions serve to sharpen the boundaries of the occupied regions, so that the map definition improves as more readings are added. The final map shows probably occupied, probably empty, and unknown regions. For positional update as well as recognition of previously mapped areas, two sonar maps by convolving them can be matched. Therefore, the method gives the displacement and rotation that best brings one map into registration with the other, along with a measure of the goodness of the match. Fig.3.21 shows a typical two-dimensional sonar map obtained using the method mentioned above. They are very useful for navigation and landmark recognition. In Fig. 3.21 the outline of the room and the major objects is shown, as well as the positions of the robot from where the readings were taken.

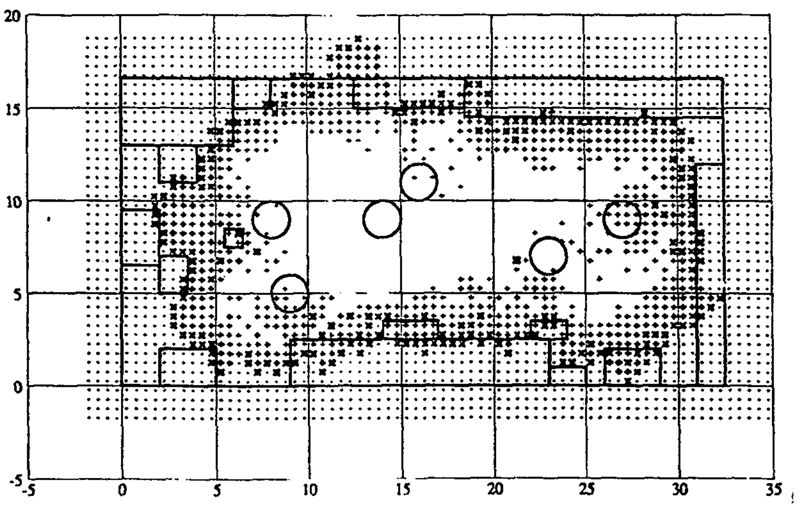


Figure 3.21 Two dimensional sonar map

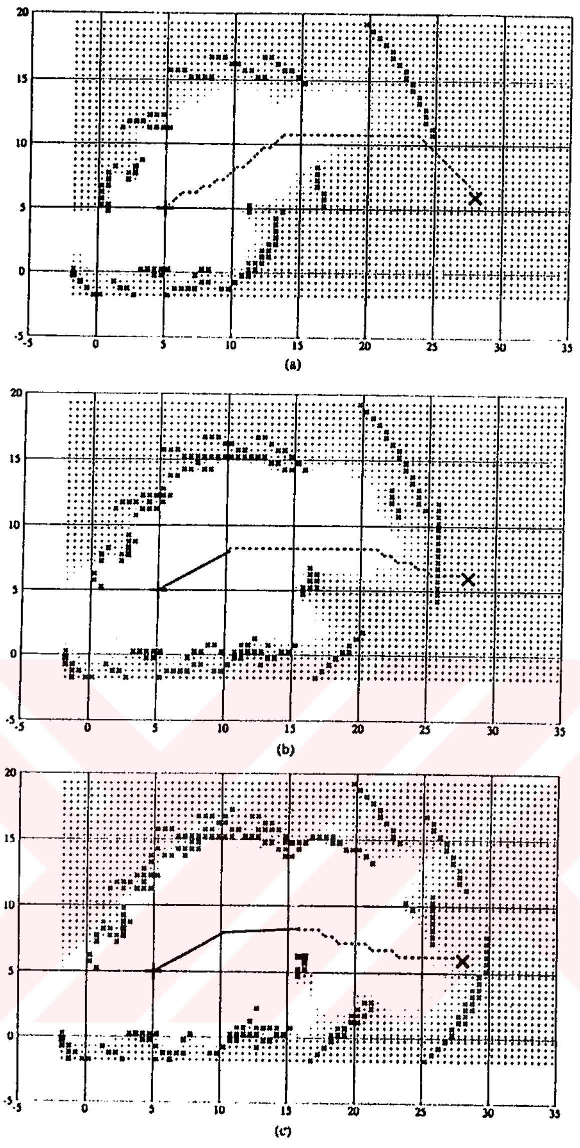


Figure 3.22 An example run of an mobile robot using sonar-based navigation

In Fig. 3.22, an example run is given. The sequence of maps shows how the sonar map becomes gradually more detailed and how the path is improved as more information is gathered. The example corresponds to an indoor run carried out in Mobile Robot Lab. A distance of approximately 7.62m was covered; the grid size is 15.24cm. Planned path is shown as dotted line, and route actually follwoed by robot as solid line segments. Starting point is solid + and goal, solid x.

If the robot is given a map of the area to begin with and only requires to keep track of its own position in the area as it moves, the mapping task becomes much easier. Ultrasonic systems can more easily handle the navigation update problem. Fig 3.23 shows a typical sonar map of an inside corner and outside corner and demonstrates the results of line fitting.

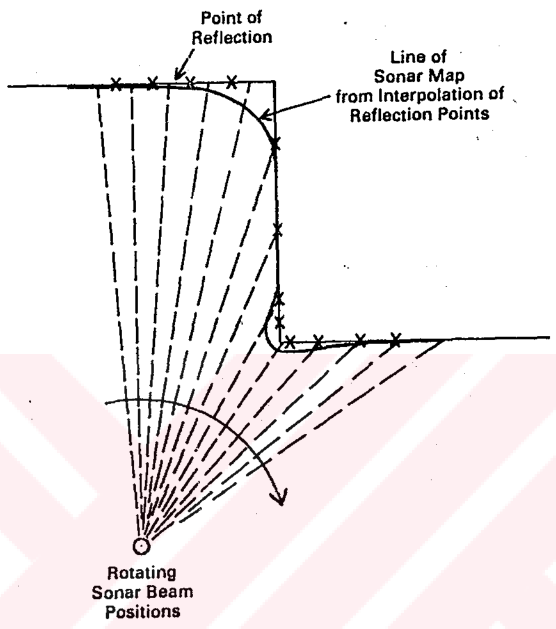

Point of Reflection
Line of Sonar Map from Interpolation of Reflection Points
Rotating Sonar Beam Positions

Figure 3.23 Mapping corners using ultrasonics

The locations of all sonar returns are shown by X's. the inside corner is cut short, and the outside corner is extended [2]. Given the recursive line fitting the corner location will still be off approximately 5cm unless the robot has an accurate map of the area in advance and can use it to correct in sonar generated map.

3.5. Triangulation Method:

The bearing measurements of passive direction-finding systems at two or more stations or points along a mobile robot trajectory can be combined by a direction-finding location system to produce an estimate of transmitter position [45]. The transmitted signal may be received at a station by line of sight propagation. A single bearing angle may be measured at each station of the location system. In the absence of noise and interference, bearing lines from two or more stations will intersect to determine a unique location. In the presence of noise, more than two bearing lines will not intersect at a single point, as shown in Fig. 3.24.

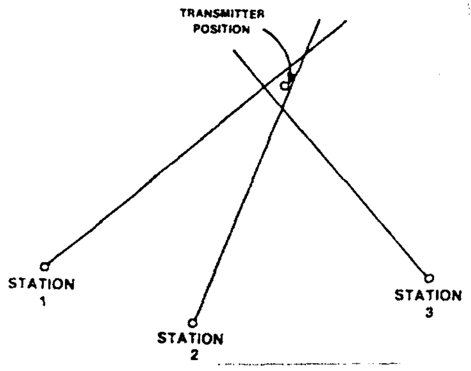


Figure 3.24 Bearing lines from three-direction-finding systems

Consequently, processing is required to determine the optimal position estimate. Let θ_i denote the bearing angle measured at station I relative to a baseline in a three-dimensional coordinate system defined so that x axis is parallel to the baseline, as it is shown in Fig. 3.25.

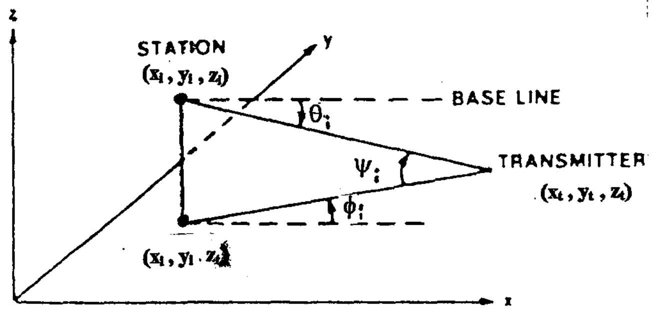


Figure 3.25 Angle definitions for direction-finding systems

If the coordinates of the station are (x_i,y_i,z_i) and the coordinates of the transmitter are (x_t,y_t,z_t), then in the absence of measurement errors, line-of-sight propagation implies that,

$$\theta_{i^*} = \cos^{-1}\left[\frac{x_t - x_i}{\sqrt{(x_t - x_i)^2 + (y_t - y_i)^2 + (z_t - z_i)^2}}\right]$$

$$0 \le \theta_i \le \pi \qquad (3.19)$$

In Fig. 3.25, the azimuth angle ϕ_i is defined in the plane passing through the transmitter and perpendicular to the z axis. It is positive in the counterclockwise direction relative to the positive x axis. If the elevation angle Ψ_i of the station relative to the transmitter is known approximately, then ϕ_I may be calculated using the geometrical relation.

$$\cos\theta_i = \cos\phi_i \cos\Psi_i \qquad (3.20)$$

If Ψ_i is sufficiently small, the measured bearing is well approximated by the azimuth, which is defined by

$$\phi_i = \tan^{-1}\left(\frac{y_t - y_i}{x_t - x_i}\right) \qquad (3.21)$$

In mobile robot application, the transmitter is known to lie on the surface of the Earth and does not have to be estimated. Equation (3.21) is used in the estimation of (x_t, y_t). The use of this equation is equivalent to the representation of the three dimensional problem by a two-dimensional model. In this two-dimensional model, the transmitter and the stations are assumed to lie in the same plane so that azimuths are identical to the bearings. Two-dimensional position estimation using bearing information is called triangulation. The beacons which are used in triangulation can be infrared, ultrasonic, laser, or even bar-codes placed at strategic positions. Although very accurate, the beacons which are used in triangulation method have a few problems. Several must be placed wherever the robot may wish to to go, thus increasing total system cost. Triangulation method also requires real-time performance. An another disadvantage is that it requires unobstructed line-of-sight which is not always possible.

3.6. Multisensory Navigation Systems:

Environment dependent sensory driven navigation is at present the key issue in mobile robots research and this is still more the case when real world applications are concerned. The combination of dead reckoning system and visual landmark detection has been studied as one potential navigation solution [46]. Dead reckoning system is moderate-priced, but not accurate enough when used alone. It needs another system to frequently correct the accumulating position error in order stay on the path. Visual landmark detection can be considered to be accurate and low-priced enough to be used to aid the dead reckoning system. The location of the vehicle can be calculated on the distance and direction of the landmarks from the vehicle. Long distances can not be measured very accurately since the measurement is based on the size of the landmark in the image. The direction of a landmark is calculated based on the position of the center of the landmark in the image. The positioning system is based on a illuminator (a halogen lamp) and camera on board the vehicle and reflective landmarks beside the route in terrain. The most important parameters of the camera used in the landmark detection are its angle of view and resolution. If the width of the road is 3m and the shortest distance at which a pair of landmarks should be seen in the image is 2m, the minimum angle of view of the camera would be 84 degrees. If the minimum size of landmark in the image is 10x10 pixels and it should be detected from a distance of 20m, the resolution of the camera should be about 1000x1000 pixels. Determination of camera parameters can be shown in Fig.3.26.

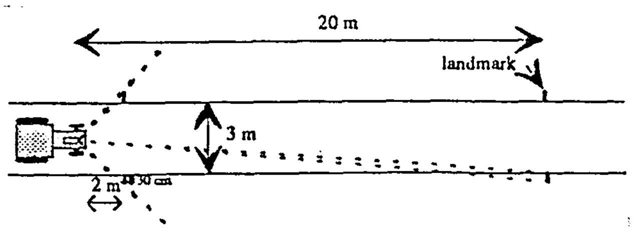


Figure 3.26 Determination of camera parameters

High reflection coefficient and retroreflectivity are the main properties of good landmark material. The landmark should be surrounded by a black border to produce a high contrast between the reflective and non reflective surfaces. Rectangle and circle can be considered as alternative shapes for the landmark. A halogen lamp is used to illuminate the terrain in front of the vehicle in such a way that the landmarks are seen as bright squares with black borders. The width of the light beam should be equal the angle of view of the camera. Therefore, the terrain is illuminated widely enough but not too widely. The landmark detection algorithm is based on the high contrast of the reflective and non reflective parts of the landmark. The search is begun at the upper left hand corner of the image advancing along the row towards the right hand of the image. If the minimum size of the landmark is determined to be 5x5 pixels, only every fifth row has to be searched. This speeds up the searching. The pixels forming the edges of landmark are searched based on the contrast. The search area is the slightly enlarged area of a potential landmark. The points of each edge are stored and a histogram analysis is performed for them. A histogram is formed from the x-coordinates of the left hand and right hand edges and from the y coordinates of the upper and lower edges of the potential landmark. If the peak of a histogram is high enough and the histogram is narrow enough, the edge is supposed to be an edge of landmark. The value of the minimum acceptable height of the histogram is relative to the quantity of the edge points found and to the estimate of the size of the landmark. If all four edges have been found based on the histogram analysis the landmark has been found. According to this approach, the position of the vehicle is calculated based on dead reckoning the accumulating error of which was occasionally reduced by detecting landmarks along the path. The Kalman filter can be used for fusing the sensor data from the dead reckoning system and the landmark detection system. A Kalman filter is simply an optimal recursive data processing algorithm [47]. It processes all available measurements, regardless of their precision, to estimate the current value of variables of interest with use of knowledge of the system and measurement device dynamics, the statistical description of the system noises, measurement errors, and uncertainty in the dynamics model, and available information about initial conditions of the variables of interest. A typical Kalman filter application can be seen in Fig.3.27.

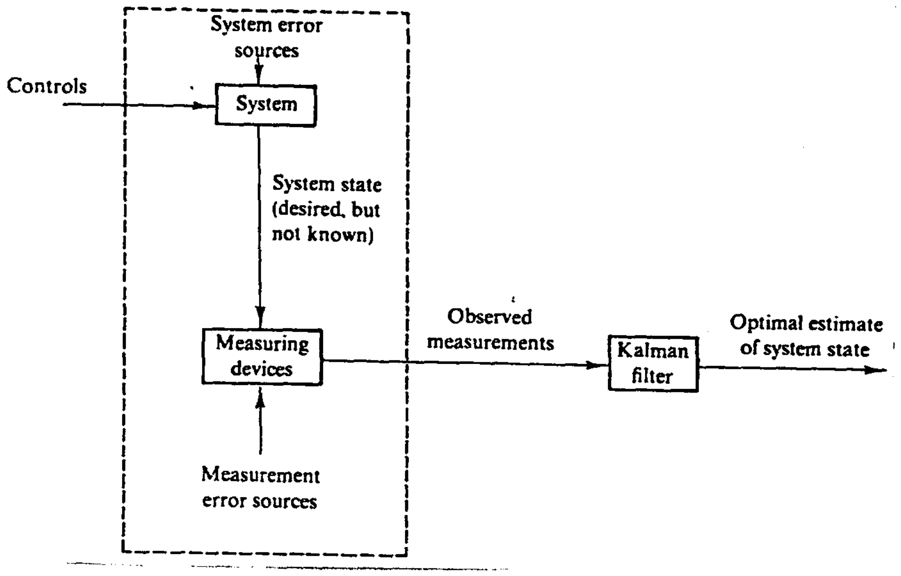

System error sources
Controls
System
System state (desired, but not known)
Measuring devices
Measurement error sources
Observed measurements
Kalman filter
Optimal estimate of system state

Figure 3.27 Typical Kalman Filter application

A system of some sort is driven by some known controls and measuring devices provide the value of certain pertinent quantities. knowledge of these system inputs and outputs is all that is explicitly available from the physical system for estimation purposes.

In this navigation system, positioning error is due to several reasons. The error is greater in the direction of the movement of the vehicle than in the perpendicular direction since the distance measurement to the landmark is based on its size in the image and the perpendicular position is calculated based on the position of the landmark in the image. The positioning error increases as the distance to the landmarks increases because in great distances one pixel of the size of the landmark corresponds to a greater range than in the shorter distances

As an another multisensory navigation system application, Hilare robot, can be given. The structure and the sensory system of the robot can be seen in Fig. 3.28. A multisensory system provides the robot with the information it needs about its environment [48]. The various sensors are used independently or in concert .

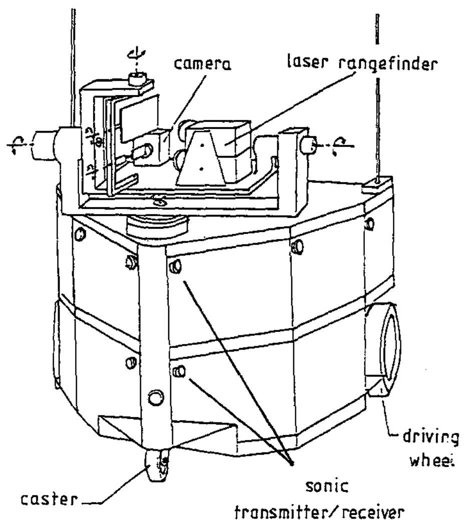


Figure 3.28 Sensory system of Hilare robot

Ultrasonic sensors: A set of 14 ultrasonic emitter-receivers distributed on the vehicle provides the range data up to 2 m. The robot uses the range data to move along an object, maneuvering to stay at a fixed distance from its surface.
Vision: A camera and a laser range-finder are the main perception system. They are mounted on a pan and tilt platform. The laser are used either in scanning mode or it measures ranges within the camera's field of using a rectractable mirror. This provides the robot with 3D data about its environment.

Hilare's position can be obtained either relatively to objects and specific environment patterns or in a constructed frame of reference. To do this, Hilare is equipped with an infrared triangulation system which operates in areas where fixed beacons are installed. The pose of the mobile robot is also computed by using the shaft encoders to integrate the robot's movements and deduce its current position knowing its initial state.

The Stanford Mobile Robot has four sensing modalities: vision, acoustics, tactile and odometry. Stereo vision, using two on-board cameras , returns the 3D location of the vertical lines within this field of view. The cameras are mounted on a pan/tilt head, giving them two degrees of freedom. The acoustic system is composed of twelve Polaroid sensors equally spaced about the circumference of the robot. These sensors return a distance that is proportional to the time of flight of an echoed acoustic chirp. To a first approximation, the acoustics find the nearest object within a thirty degree cone. Though the acoustic system has very low angular resolution, it measures depth quite accurately making it useful for guarded moves. As a last line of defense, if the vision and acoustic system miss an object, there is a tactile sensing system, composed of twelve bumpers with internal tape switches. Besides emergencies, the information can augment the world model. The bumpers are also useful for navigating through tight areas such as doorways where the edges of the door are within the minimum ultrasonic range and not within the region of stereo vision. Finally odometry determines the robot's position.

Generally, most researchers have used dead reckoning method coupled with periodic calibration using one of the above methods for navigation of mobile robots. Although this approach seems to be quite effective, more needs to be done to develop a reliable and accurate navigation system. The state-of-the-art of navigation system technology today is still at an infancy stage. For autonomous mobile robots to have practical widespread application, an accurate, reliable, fast, and cost effective navigation systems must be developed in mobile robotics field.

CHAPTER 4. MOTION PLANNING IN MOBILE ROBOTS

The general motion planning problem for a system of autonomous mobile robot can be stated as follows:

1. Given an initial state of the mobile robots,
2. Given a desired final state of the mobile robots,
3. Given any constraint on allowable motions, find a collision-free motion of the mobile robots from the initial state to the final state that satisfies the constraints.

A state of the mobile robot may include a description of such things as the position , orientation, and velocity of each vehicle. When it is assumed that the environment is static and the only constraint on the motions of the mobile robots are they do not collide, it suffices to specify collision-free paths for the mobile robots to follow. In the majority of the work in motion planning, these conditions are assumed and such work can be better described as **Path Planning**.

Briefly, the purpose of path planning is to determine a collision-free path for a mobile robot that moves in a workspace including obstacles. The problem is simpler than dealing with an industrial robot when dealing with a mobile robot, as the problem has only 3 degrees-of-freedom. However, other constraints have to be considered, in particular the usability for the robot to turn without moving forward and backward. Position and orientation can not be considered separately. The path planning problem can be divided into two sections. These are Global Path Planning (GPP) and Local Path Planning (LPP).

4.1. Global Path Planning:

Global path planning requires a pre-learned model of the domain which maybe a somewhat simplified description of the real world and might not reflect recent changes in the environment, namely, in global path planning the robot must determine where it is and where its goal is. This determination is normally done in relation to a global map of the area that the robot has in its memory. Once its current position is updated on the map, the robot can plot a path between itself and its goal. If no obstacles are present, plotting the path is straightforward. With obstacles, the robot must use a path planning algorithm to determine the shortest route to its goal. Once the path has been updated, the robot can move forward. During this motion, the robot typically follows three steps:

1. It turns to correct the heading between itself and its goal.
2. It moves toward its goal a small distance, depending on the accuracy of its map, its sensors and the turn.
3. After this set is traversed, the robot again computes where it is and where its goal is and plots a new path to the goal, correcting any small errors that occurred in the first move. Since the calculations take time, there could be series of starts and stops.

4.2. Local Path Planning

In local path planning, the primary purpose is to find the best way around unexpected obstacles. Local path planning adjusts robot motion as necessary to handle unexpected obstacles that the robot finds blocking its path. Briefly, it is tactical, that is, carried out as a reaction to an unexpected event. Whereas global path planning may operate on a pre-stored model, local path planning requires a model that reflects the state of the environment, including changes, as the plan is being executed.

Local path planning is usually more accurate than global path planning because the available sensor resolution cover a much smaller area. It is usually necessary to be more accurate since precise movements must be made. Since the resulting local path planning will always be reasonably short, the first solution can always be used and the robot need not optimize it. If the robot can find no short-range solution to its problem, local path planner turns the problem over to the global path planner to see if it can determine an alternative route to the desired goal.

4.3. Collision Avoidance

Three types of obstacles typically found in a real-world environment: These are stationary / static, stationary / dynamic, and moving obstacles. Stationary / static refers to things like walls or factory workbenches, which are always present at the same position. Stationary / dynamic refers to objects that are stationary when seen by the robot, but whose position could have changed from the last time it was seen, such as open or closed door or an office chair that had been moved. Moving objects refers to objects that are usually in motion, such as people.

Collision detection and avoidance are two of the four requirements imposed on any mobile robot that is going to move autonomously. It must know where it is and where it wants to go, which is Global Mapping. It must know how to reach the goal, considering known obstacles in the way, which is Global Path Planning. Next as it travels, it must recognize additional, unexpected objects in its path to prevent collisions, which is Collision Detection. Finally, it must navigate, around these obstacles to continue toward its goal, which is Local Path Planning.

The first goal of a collision avoidance is to ensure that the mobile robot does not contact an object because of errors in its navigation or because a new object has suddenly appeared in its path. When the robot finds its direct path to a goal blocked, it must find a method to circumvent the blocking object, if possible, by using some type of local path planning. A collision detection system by itself can alert the robot to a potential collision. Separate software is needed to provide a local alternative path. Collision detection systems must operate in real-time while the robot is in motion.

The type of collision-avoidance system required depends on the rules under which the robot is operating. A mobile robot generally travels under one of four conditions:

1. Fixed Paths : The robot is following a fixed path, such as a burried wire that is implemented in AGVs, in which the only action the robot should take on detecting something to stop.
2. Known Environment : Following a fixed, predetermined path where collision avoidance will be used to move the robot temporarily from this path and then return to it.
3. Changed Environment : Following a predetermined path between two locations that will vary between runs because the robot can determine a better path based upon its own map of the area or because changes occur in the environment. Examples of changes include doors opened and closed or movement in goal point location.
4. Unknown Environment: Exploring a new environment which no current map exists and the robot must explore and prepare its own map. For this condition, the goal might be considered to be a complete map of the area. In this case, the collision detection system will only alert the robot to a potential collision. The global mapping would have to determine alternative paths.

In Helpmate autonomous mobile robot , obstacle avoidance is based on the composite local map [49]. Data from the sensor subsystems is rationalized and placed on the map. The map is scanned in front of the robot to see if there are any obstacles. If there are, the robot slows or stops, depending on the distance to the perceived object, and confirms the presence of the obstacle. If the obstacle has moved, as a person will, the robot waits for the path to clear and moves on. If the object is stationary, the robot plans course around the obstacle, sensing the extent of the obstacle as it moves around it and stays a minimum distance away.

Some of many reasons for the lack of a complete and collision avoidance system are sensor limitations, system time, and computing constraints and path geometry considerations. Collision detection sensors suffer from three types of problems:

1. In sufficient coverage because of the cone pattern found in most obstacle detection systems, there will always be areas near the robot that the sensor can not see. Multiple sensor integration can solve some of this problem.
2. A second problem is to determine the exact location of an obstacle and the exact width of an available opening. Sensor inaccuracies often add centimeters to the required clearance distance that the robot must traverse. Another problem is found when attempting to determine an alternative route around the obstacle. The limited coverage of the sensors does not readily support side views and the placement of the obstacle often blocks part of the optional path from view. With the mobile robot traveling at a reasonable speed; it may not have enough time to detect a potential collision and react to it unless a separate computer is assigned the task of collision detection and perhaps subsequent path planning.
3. A third problem is due to the three-dimensional problems inherent in collision avoidance, while a two-dimensional approach is often taken for path planning. The robot travels in two-dimensions, and one approach often used for global path planning is to shrink the robot to a point and extend the boundaries of all obstacles by the radius of the robot , which is called Configuration-Space Approach [50]. This approach allows fewer complex calculations for determining which path will permit the robot to clear any obstacles. Unfortunately this procedure is usually insufficient in a collision avoidance situation, unless the robot and objects that may collide with it are smooth and symmetrical. Although acceptable for a research environment, this situation is not what one would expect in an industrial setting. Objects may hang down from a ceiling, protrude from a wall, or rise from the floor, and yet be so placed as to not be seen from any collision -avoidance sensor.

4.4. Path Planning Techniques

Path Planning with complete information is generally formulated as follows. Given a solid object, in two-or three-dimensional space of known size and shape, its initial and target position and orientation and a set of obstacles whose shapes, positions and orientations in space are fully described [51]. The task is to find a continuous path for the object from the initial position to the target position while avoiding collisions with obstacles along the way. It is assumed that the surfaces of the moving object and of the obstacles are algebraic. Because full information is assumed, the whole operation of path planning is a one-time, off-line operation. The main difficulty is not in proving that an algorithm that would guarantee a solution exists, but in obtaining a computationally efficient scheme. Conceptually, cases of arbitrary complexity can be considered, given the fact that a solution is always feasible. The advantage of dealing with complete information is that any optimization criteria such as the shortest path or the minimum-time path or the safest path can be easily introduced. In path planning with complete information approach, some approximations are made in the algorithms. A slight change in the specified accuracy of the approximation can cause a dramatic change in the approximated surfaces and eventually in the generated paths. Some of the path planning techniques which can be used in global or local path planning will be presented in the following section.

4.4.1. Configuration-Space Approach:

A configuration of a part is a set of parameters which uniquely specify the position of every point on the part, and configuration space is the set of all possible configurations. In configuration space the problem of planning the motion of a part through a space of obstacle is transformed into an equivalent, but simpler problem of planning the motion of a point through a space of expanded configuration space obstacles. Expanded obstacle is the result of special transformation which is done with all obstacles of the world representation in order to enable a point-wise representation of the mobile robot [50]. The actual dimensions of the obstacle are being increased in such a way that the walls of expanded obstacle would rather represent the trajectory of the robot moving along the wall without damaging it.

The configuration space can be two dimensional or three dimensional depending upon whether or not the mobile polygon is allowed to rotate [52]. Suppose triangle A is the mobile robot Fig. 4.1 and rectangle B is an obstacle.

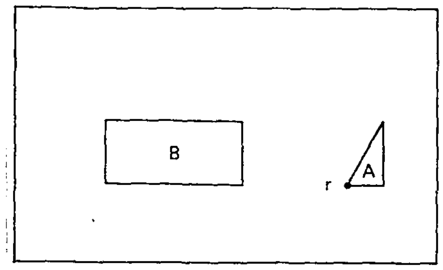


Figure 4.1. A scene with two polygonal parts

It is assumed that A can translate but not rotate. Given a mobile robot A, a reference point r somewhere on the mobile robot is chosen. To convert the configuration space, the mobile robot is shrunk to its reference point r while simultaneously growing the obstacles to compensate. In the case of pure translation of the mobile robot A, this growth is achieved by sliding part A along the boundary of obstacle B and tracing out the locus of points traversed by the reference point r. As triangle A is shrunk to its reference point r, rectangle B grows into five sided polygon, B_A. Expanded obstacle B_A refereed as a configuration space obstacle induced by A. When the original polygonal scene in Fig 4.1 is redrawn in the configuration space generated by triangle A, the result is as shown in Fig 4.2 .

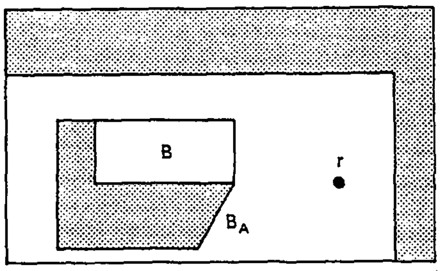


Figure 4.2 Configuration-space induced by part A

As long as the reference point of triangle A stays outside the configuration space obstacle B_A, triangle A will not collide with obstacle B. Thus, the original problem in Fig. 4.1 of finding a path for a triangle has been converted into the equivalent problem in Fig. 4.2 of finding a path for the point r. It should be noticed that, the configuration-space obstacle B_A in Fig. 4.2 contains the original obstacle B as a subset. furthermore, the mobile part A and the obstacle B are both convex and so is the configuration-space obstacle B_A. computing configuration-space obstacles is a key step in solving the path planning problems.

Let A and B be convex and let B_A be the configuration-space obstacle generated by a using reference point r. If n_A, n_B, n_{BA} denote the number of vertices of A, B, and B_A, respectively, then:

1. $n_B \le n_{BA} \le n_A + n_B$
2. B_A is convex
3. if $r \in A$, then $B \subseteq B_A$.

Thus the number of vertices in the configuration-space obstacle is bounded from below by the number of vertices in the original obstacle, and from above by the number of vertices in the original obstacle plus the number of vertices in the mobile part. Let A be a convex mobile polygon with reference point r and let $\{B^k : 1 \le k \le m\}$ be a convex polygonal obstacles. Suppose s and g are start and goal points, respectively, for the reference point r. If polygon A does not rotate, then the shortest path from s to is a piece-wise linear path $\{r^k : 1 \le k \le n\}$ whose interior points ($1<k<n$) are vertices of configuration-space obstacles $\{B_A^k : 1 \le k \le m\}$.

Therefore as long as the mobile part A does not rotate, the shortest path from the start s to the goal g is a piece-wise linear path whose breakpoints are vertices of the configuration space obstacles induced by A. Since there are a finite number of vertices, they can be efficiently searched for the shortest path using graph theory techniques. For example, in Fig 4.3-a , the start and goal values of the reference point are denoted as s and g, respectively, are shown.

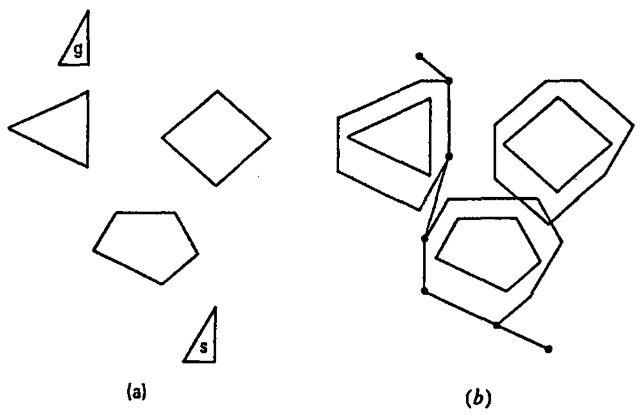


Figure 4.3. Finding the shortest path using configuration-space

In Fig 4.3-b, two configuration obstacles overlap, which means that there is insufficient space for the mobile part to pass between them. The shortest path from s to g is also seen. The path in Fig 4.3 has the mobile polygon sliding along the edges of the obstacles. This is characteristic feature of the shortest path, a feature which can cause practical problems if there is uncertainty in position, orientation, size or shape of the polygons. To avoid the possibility of a collision during the sliding operation, a thin buffer region can be placed around the mobile polygon before generating the configuration space obstacles, which will be slightly larger. When this is done, the path will no longer be optimal in the sense of the shortest path, but it will near the optimal if the buffer region is kept small.

Configuration-space is very effective when applied to purely translational motion in a plane, since the shortest path can readily be found . However, in many instances the mobile object must be rotated if it is to reach the goal. when the mobile polygon is allowed to rotate, the configuration-space obstacles are no longer polygonal. Instead the surfaces of configuration-space obstacles are curved in the orientation dimension. A simple way to handle rotations is to employ an enlarged polygon which encloses the mobile part the mobile part over a range of orientations as shown in Fig 4.4.

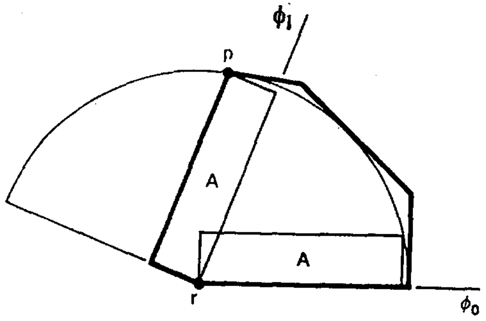


Figure 4.4. Enclosing the rotating part in convex polygon

The mobile polygon A is allowed to rotate about reference vertex r over a range $[\phi_0,\phi_1]$. If the distance from the reference vertex to the farthest point p on the mobile polygon is d, then the rotated polygon will be contained in a circular sector of radius d and angle $\Delta\phi$, where $\Delta\phi \geq \phi_1 - \phi_0$. The sector can be truncated at the leading and trailing edges of A and the truncated sector can then be enclosed by a polygon as shown in Fig 4.4

The virtue of this approach is that it converts the problem of finding a path for an enlarged part that only translates. If a path is found, then the mobile part can assume any orientation in the interval $[\phi_0,\phi_1]$ at each point along the path. The drawback of the simple technique is that it is possible that a path from s to g may not be found when one in fact exists. To address this problem, the three-dimensional configuration-space $\{x,y,\phi\}$ must be searched directly. One technique is to construct a sequence of configuration-space slice projections, that is, configuration-space obstacles corresponding different value of ϕ. Movement within a slice is a pure translation, while movement between slices can be achieved with a pure rotation. The three dimensional configuration-space obstacles can be approximated using a set of slice projections and the remaining free space can then be searched for a collision-free path.

The advantages of C-space description of the workspace are:

1. The computed path can be used directly for the robot as no further computation is necessary.
2. The configuration-space defines the configuration unambiguously.

However there are also some disadvantages:

1. The configuration-space must be calculated in advance, a process which even fast implementations take relatively long time.
2. To reduce the time to create configuration-space, most algorithm use some kind of conservative approximation, thus some paths might not be found in configuration-space search.
3. Higher dimensional configuration-space uses a lot of memory especially with a fine resolution.

4.4.2. Freeway Approach:

An alternative to the configuration-space approach to solving the path planning problem is to search the free space directly. Brooks [53] offered an explicit representation of free space based on overlapping generalized cones having straight spines and nonincreasing radii. These representations of the space between obstacles are called " freeways ". Translations are performed along freeways and rotations are performed at the intersections of freeways.

Generalized cones are a commonly used representation of volume for modeling objects in artificial intelligence computer vision systems. A generalized cone is formed by sweeping a two-dimensional cross section along a curve in space, called a spine, and deforming it according to a sweeping rule. Two dimensional generalized cones generated by two obstacles and workspace boundary can be seen in Fig 4.5.

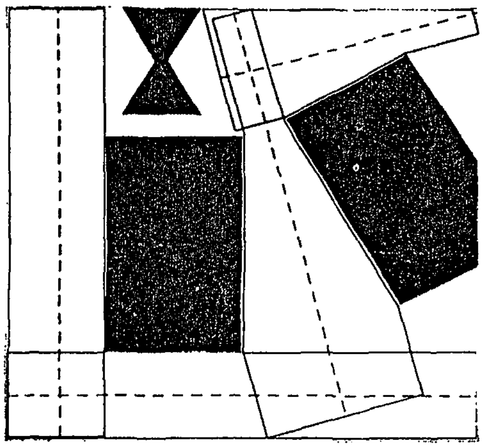


Figure 4.5 Generalized cones generated by two obstacles

The spines are straight and the cross sections are line segments held perpendicular to the spine with left or right radii. The sweeping rule will independently control the magnitudes of the left and right radii as piece-wise linear functions. Fig 4.6 also shows the complete representation used for cones describing parts of freespace.

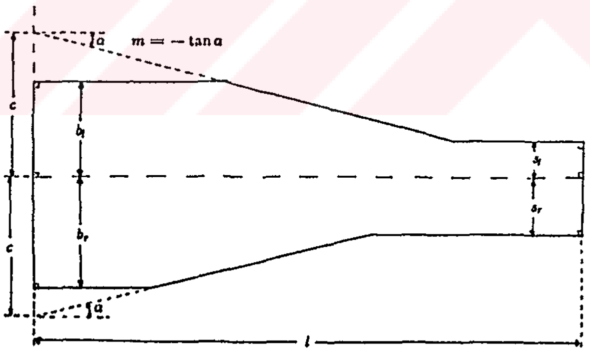


Figure 4.6 The structure of a generalized cone

There is a straight spine, parameterized over the $t \in (0,l)$ where l is the length of the cone. If the sides of the cone are not parallel to the spine, then t=0 corresponds to the wider end. On both the left and right, the maximal radii achieved over the length of the cone occurs at t=0, are denoted b_l and b_r, respectively, describing the big end of the cone. The minimal radii achieved occur at t=l and are denoted s_l and s_r, describing the small end of the cone. If $b_l = s_l$ and $b_r = s_r$, then the two sides of the cone are parallel to the spine. If not, then there is a symmetric thinning of the cone, where the left and right radii of the thinning parts of the cone are both given by the expression mt+c where m and c are constants and generally $m \leq 0$, $c \geq 0$. The seven constants l, b_r, b_l, s_r, s_l, m and c completely specify the shape and the size of the generalized cone. In addition, its location and orientation must be determined concurrently with computing these parameters. The algorithm to find a collision-free path proceeds as follows:

Step 1. The generalized cones which describe free space are computed.

Step 2. The cones are pairwise examined to determine if and where their spines intersect. Up to this stage, the algorithm is independent of the object A to be moved through the environment. Each spine intersection point for each cone is annotated with the subset of $[0,2\pi]$ which describes the orientations for the object A in which it is guaranteed to be completely contained in the generalized cone.

Step 3. The final stage of the algorithm is to find a path from the initial position to the goal position following the spines of the generalized cones, and changing from cone to cone at spine intersection points. If the object is kept at valid orientations at each point on each spine, then the path will be collision-free. In Fig 4.7, a path found by freeway algorithm can be seen.

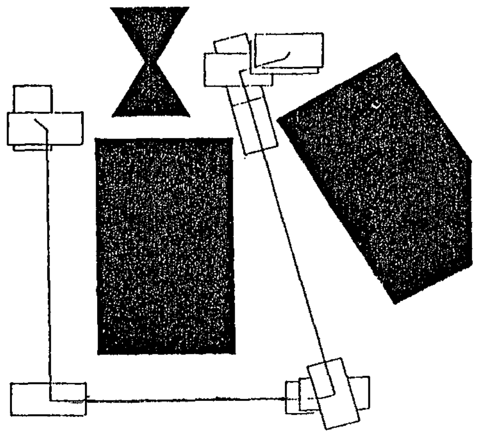


Figure 4.7 Path found by freeway approach

The advantages of freeway approach are as follows:

1. The path found and the amount of computation needed, are completely independent of the chosen world coordinate system.

2. It generates paths for the mobile robot that stay away well from the obstacles, although this means that the paths are somewhat longer than the shortest path.

3. When the workspace is sparsely populated with obstacles, this method is very fast and quite effective.

The principle drawback of this approach occurs when the workspace is cluttered with closely spaced obstacles. In this condition, the approach sometimes fails to find a safe path when one exists.

4.4.3. Generalized Voronoi Diagrams (GVD):

A refinement of the freeway method can be achieved by using an explicit representation of freespace based on a Generalized Voronoi Diagram. A Generalized Voronoi Diagram is defined as a subset of free space with each point simultaneously closest to two or more obstacles. The edges represent basically the maximum clearance between the obstacles and are thus intuitively the safest way to pass them. The Voronoi diagram is a strong deformation retract of free space so that free space can be continuously deformed on the diagram [54]. This means that the diagram is complete for path planning. Searching the original space for paths can be reduced to a search on the diagram. Reducing the dimension of the set to be searched usually reduces the time complexity of the search and also the diagram leads to robust paths. The space outside the obstacles can be continuously deformed on to the diagram. To find a path between two points in free space, it suffices to find a path for each point on to the diagram and to join these points with a path that lies wholly on the diagram.

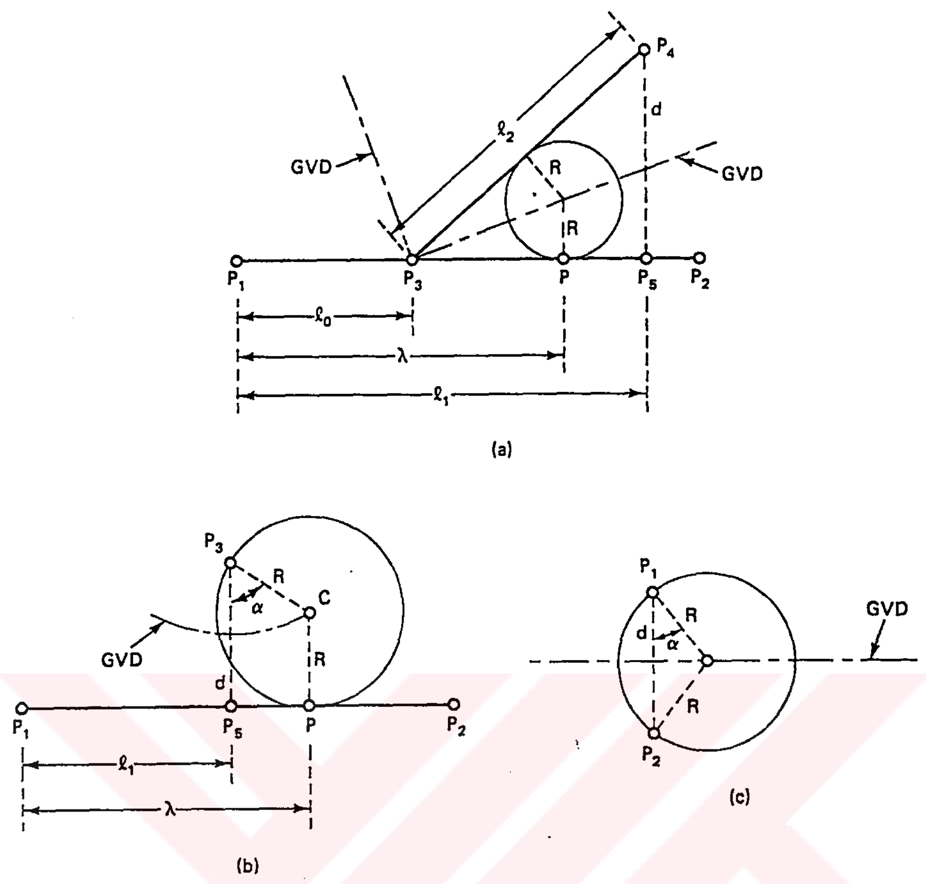


Figure 4.8 Three basic types of interaction in a GVD

In constructing a GVD, there are three basic types of interaction, as shown in Fig 4.8. The first type interaction is between a pair of edges, as shown in Fig 4.8-a, the second type of interaction is between a vertex and an edge, as shown in Fig 4.8-b. The third type of interaction is between pair of vertices, as shown in Fig 4.8-c. More complex generalized Voronoi diagrams [52] are constructed using combinations of the three basic GVD types .

A complete GVD can be seen in Fig 4.9. In order to conduct an efficient search, the GVD is first converted to an equivalent graph of nodes and arcs.

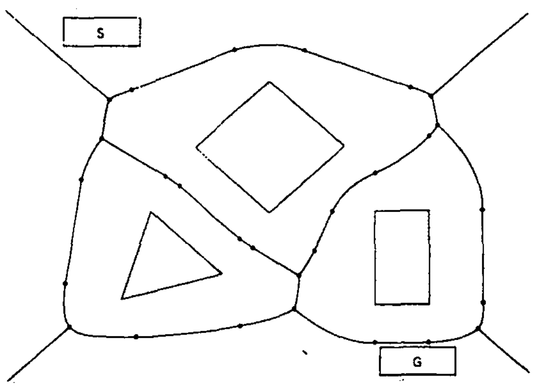


Figure 4.9 A complete GVD

There are three types of nodes that arise in the GVD graph. First junction nodes, which are generated when three or more GVD lines intersect; second terminal nodes, which correspond to dead ends of GVD lines; third pseudo-nodes, which are start and goal nodes inserted in the graph near the start and goal points of the mobile robot. Pseudo-nodes serve as entry and exit points on the graph. The arcs of the graph are GVD lines connecting pairs of nodes. An arc consists of a sequence of piecewise-linear sections between via-points, each via-point being characterized by the parameters (x,y,R), where (x,y) are the coordinates of the via-point and R is the radius of the GVD at the via-point. Smooth parabolic GVD curves can be approximated arbitrarily closely with piecewise-linear arcs by controlling the spacing between the via-points. To search for a path from the start node to the goal node, one starts by examining the nodes adjacent to the start node. Nodes adjacent to these nodes are then investigated, and the search continues to expand all branches of subgraphs until each subgraph reaches a dead end or the goal node. If the minimum radius of an arc is found to be less than half the width of the mobile robot, a collision is inevitable. Consequently, these narrow arcs are regarded as dead ends. When a node adjacent to subgraph A is already reached by subgraph B and the total length for subgraph A exceeds the length for subgraph B, subgraph A is discarded. The shortest subgraph which reaches the goal node is taken as the candidate for the Shortest collision-free path for the mobile robot. An example of a path found by this method can be seen in Fig 4.10.

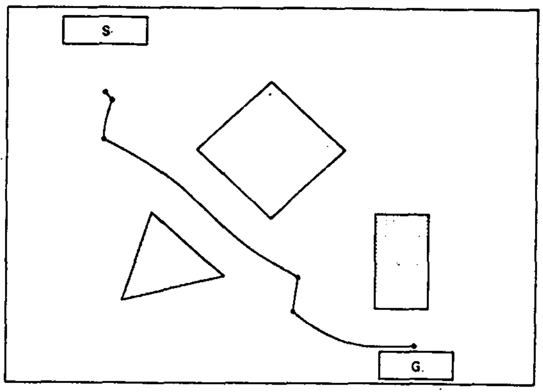


Figure 4.10 The shortest GVD path

To simplify the motion heuristics, the mobile robot can be assumed to be a rectangle. This is not a severe restriction for convex polygonal parts. Indeed, if the mobile robot is approximately circular in shape, then the path planning problem simplifies, because the orientation of the mobile robot is no longer an important factor. Thus the more interesting case occurs when the mobile robot is elongated, and it is here that a rectangular hull is often an effective approximation of the actual object size and shape. The GVD technique computes rotations quite efficiently and therefore generate fast run times, as in the Freeway approach. It should be noted that the final motion is quite smooth, in the sense that it follows curved GVD lines, stays well away from the obstacles when possible, and rotates as it translates, not just at isolated points in the workspace. A path planned in a cluttered environment can be seen in Fig 4.11

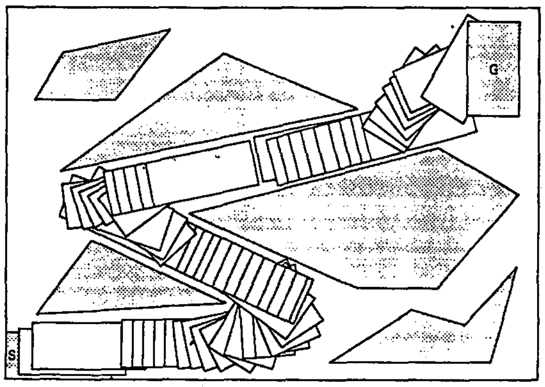


Figure 4.11 A path planned in a cluttered environment

4.4.4. Cell Decomposition Approach:

Configuration-space can also be searched by decomposing it into a grid of cells with each cell marked empty, full or mixed depending upon whether or not a configuration-space obstacle is present [55]. A cell is full if it is completely within an obstacle, empty if the cell does not intersect an obstacle, and mixed otherwise. Generally an "adjacency graph" is created with nodes corresponding to empty cells and arcs between two nodes if these two nodes are adjacent. In the next step, the graph is searched using a heuristic search algorithm to limit the search. Usually as a search algorithm the A^* algorithm or a modified version is used . This process is repeated until a path is found through empty cells. In A^* algorithm [56] first an heuristic to estimate the distance from a given node to the goal node is estimated. Adding this to the distance traveled or cost incurred to reach the node, an estimate of the total length of the path currently being explored is obtained. Then on those paths are worked that, at any given point in the search, have the shortest estimated path length. Further, it can be shown that if the distance to the goal is consistently underestimated, we cannot help but find the shortest or least costly path. To implement an A^* search:

1. Form a one-element stack consisting of a zero-length path from the root node to nowhere .
2. Cycle through steps a-c until the stack is empty.
    a) Test the first path to see if it leads to the goal.
    b) Exit with success if the tested path leads to the goal.
    c) If the first path does not lead to the goal:
        1. Remove the first path from the stack.
        2. Form new paths from the removed path by expanding one level.
        3. Add the new paths to the stack
        4. Sort the entire stack by the sum(cost incurred so far+underestimate of the cost remaining), least cost on top.
        5. If there are multiple paths to a node, retain only the path with the lowest cost to that node.
3. This Exit with failure if no solution is found.

In this method, if one exists to find a path is guaranteed, but in a very cluttered environment, the size of the cells might become very small in order to find a feasible path and this makes the search complex so the computational effort can sometimes be considerable, particularly for a real time implementation.

The simplest cell decomposition method is to create equisized cells. It has the advantage of quick and easy cell creation and intuitive cell adjacency, but a huge number of cells is created which causes the search to be exhaustive. In order to make the search better, another representations were proposed. One of them is to allow big cells in uniform space (either empty or full) and smaller cells only when needed for example at the border of an obstacle. This approach limits the number of cells thus the search graph, but complicates the adjacency graph. The most straightforward way is to use a quadtree or octree method to represent the space. Quadtree can be defined as " a method of hierarchical self-referencing elements of a grid which is based upon on the ide of simultaneous existence of a multiplicity of grids with different resolution and with capability of referencing the cells of the lower level grid to the cells of the upper level grids. Quadtree structure can be seen in Fig 4.12.

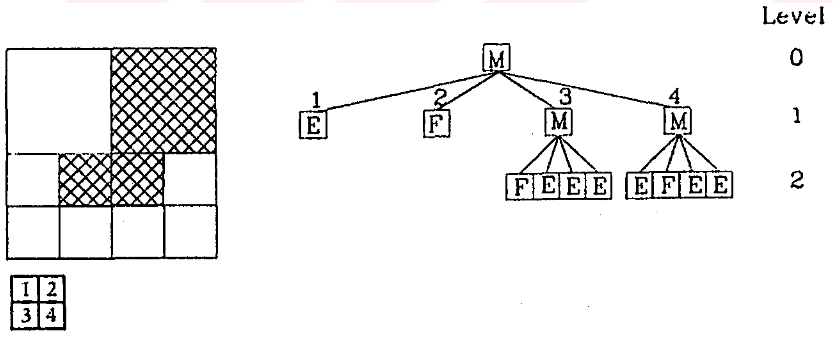


Figure 4.12 Quad-tree structure

The tree structure makes the cells easily representable, but tends to produce a high number of cells as mixed alls are often divided into the other mixed cells, some mixed cells can be cut in such a way that maximize either the empty or full area, but adjacency graph gets complicated. Decomposition of the area and adjacency graph can be seen in Fig. 4. 13.

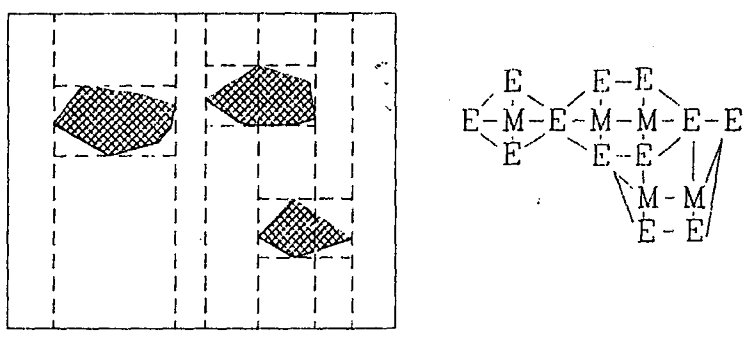


Figure 4.13 Decomposition and adjacency graph

Cell decomposition technique has been used for the mobile robot Hilare. The cycle starts from the analysis of the obstacles map [10]. The empty areas between obstacles are tesselated into convex polygonal cells. Each cell boundary has associated with it an entry of exit segment. A search graph is generated from the cell to cell boundary segments and it is this graph which is searched for the path solution based on the shortest path length as it seen in Fig. 4.14.

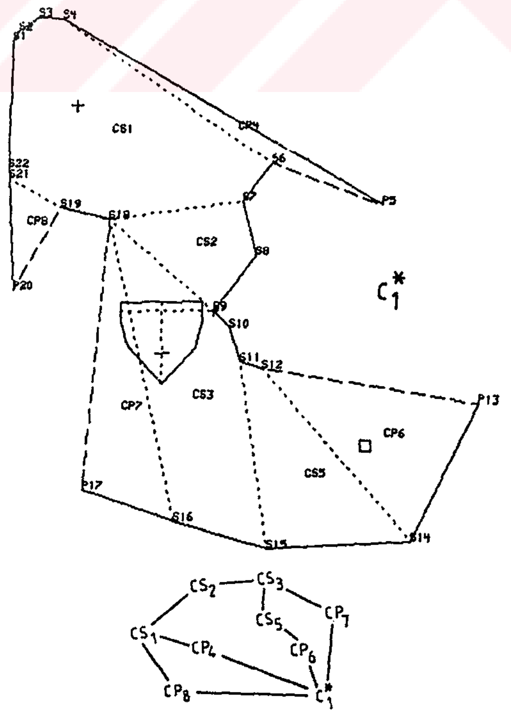


Figure 4.14 Cell structuring and adjacency graph for Hilare

4.4.5. Hypothesize and Test Approach:

Hypothesize and Test method is the first realized method to solve the automatic path generation problem. In this method, a path is proposed which connects the initial position with the final position. In the next step, this path is tested for collisions with obstacles.

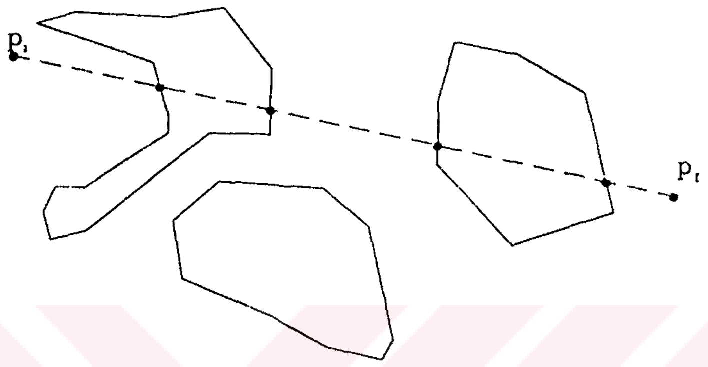


Figure 4.15 Hypothesize and test approach

If the path is not collision-free, the obstacles which caused the collisions are examined and a different path is proposed using applicable heuristics. Fig 4.15 shows the illustration of hypothesize and test approach. Generally the first proposed path is divided into segments in which collision occured are changed. Especially in a cluttered environment, it can extremely difficult to propose a new path, as a change often will result in a collision with a different obstacle. Hypothesize and Test algorithms are generally simple to implement and can find a path very fast in an uncluttered environment, but are inefficient for a cluttered environment.

4.4.6. Potential Field Approach:

During the past years, Potential Field Methods for obstacle avoidance have gained increased attention from researchers in the field of robotics and mobile robots. The idea of imaginary forces acting on a robot has been first suggested by Khatib[57].

This method was presented in the context of manipulator collision-avoidance. Its application to mobile robot is straightforward. The philosophy of the artificial potential field approach can be described as follows:

" The manipulator moves in a field of forces. The position to be reached is an attractive pole for the end-effector and obstacles are repulsive forces for the manipulator parts."

The algorithm creates a potential field with a minimum at the final position and superposes this field with positive fields created by obstacles. The resulting artificial potential field as it is shown in Fig 4.16 must be differentiable as the path is usually searched by the following the flow of the negated gradient vector field generated by this potential field.

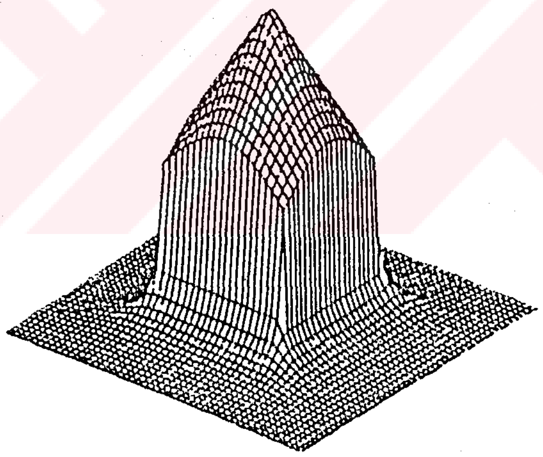


Figure 4.16 Potential field of a 2-D square

The potential around the obstacles is usually inverse proportional to the distance to the obstacles. To simplify the distance calculation, the robot is usually approximated as a set of points.

In APF, path planning is made by solving the following differential equation:

$$M(x)dx/dt^2+F_c= - grad(U_g(x)) \qquad (4.1)$$

with :

$M(x)$:	Inertial matrix
F_c:	Centrifugal Force
U_g:	$U_a(x)+U_r(x)$
U_a:	Attractive APF (goal)
U_r:	Repulsive APF (obstacles)

The trajectory of the mobile robot tends to follow the maximum negative slope between its starting point and its goal which is "deep" value of the gradient [58], avoiding obstacles which are represent by peak values of the gradient as it is shown in Fig 4.17.

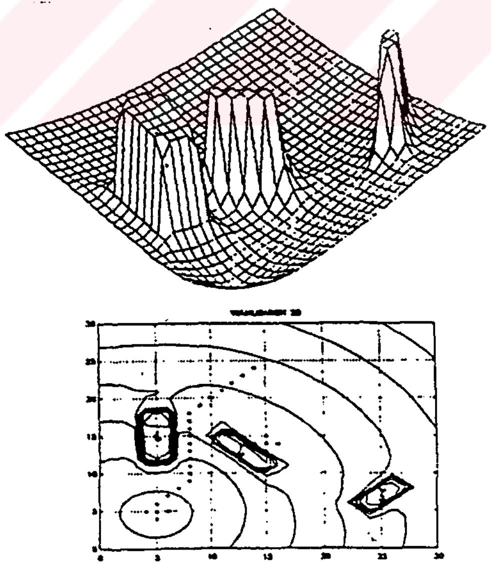


Figure 4.17 Example of global artificial potential field, (APF)

Mainly four significant problems that are inherent to potential field methods. These are:

1. Trap situations due to local minima: Local minima problem is the best known problem with PFMs. A trap situation may occur when the robot runs into a dead end. For example, inside a U-shaped obstacle. Trap situation can be resolved by heuristic or global recovery. . Generally, trap situations that are remedied with heuristic recovery rules are likely to result in "non-optimal" path. In global recovery, the local path planner monitors the robot's path, when local minima is detected, the global path planner is invoked to plan a new path based on the available information [59]. To overcome the local minima problem, the navigation function which is a potential field without local minima has been also proposed. But the analytical description of the transformation is complicated and requires precomputation.

2. No passage between closely spaced obstacles: Under artificial potential field, the mobile robot does not pass among densely spaced obstacles. With PFMs, the repulsive forces from the obstacles 1 and 2 are combined into the two lumped repulsive forces F'_{r1} and F'_{r2} respectively. The sum of all repulsive forces points straight away from the openning between the two obstacles. As it is shown in Fig. 4.18, depending on the relative magnitude of the target-directed force F_t, the robot will either approach the opening further, or it will turn away.

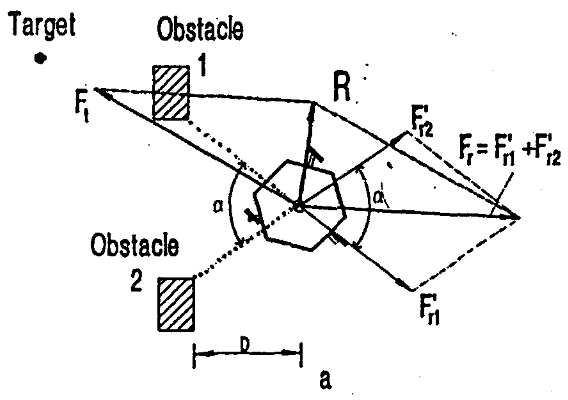


Figure 4.18 Under APF, no passage between closely spaced obstacles

3. Oscillations in the presence of obstacles: PFMs has tendency to cause unstable motion in the presence of obstacles. The stability condition is proportional to the

robot speed, V, and the time constant τ of the steering motor. It states that for each set (V, τ, n, F_r, F_t), there exists an angle α at which unstability occurs Fig. 4.19.

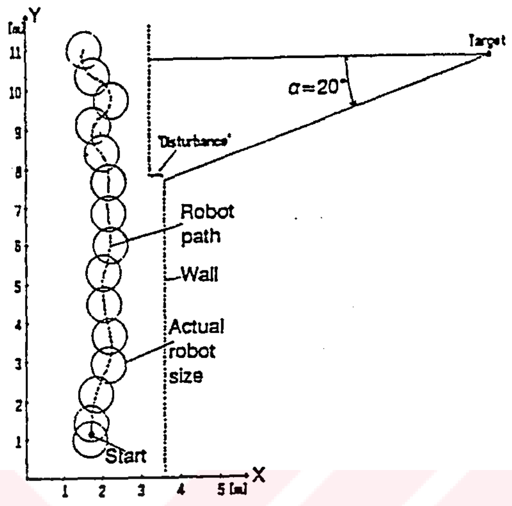


Figure 4.19. Motion of a mobile robot in the vicinity of an obstacle

The angle α depends on the position of the robot relative to an obstacle in the space configuration. In Fig 4.19, the actual position of the robot at a specific time intervals is indicates by circles. the oscillation that commence as the robot reaches the discontinuity at angle α should be noticed.

4. Oscillations in narrow passages: A similar yet more severe problem with PFMs occurs when the robot travels in narrow corridors, in which the robot experiences repulsive forces simultaneously from opposite sides and unstable oscillation occur.

A kind of PFM, which is called Virtual Force Field (VFF) has been developed by Borenstein and Koreny [60]. It was originally designed for real- time obstacle avoidance with fast mobile robots. The VFF method allows continuous, smooth, and relatively fast motion of the controlled vehicle among unexpected obstacles and does not require the vehicle to stop in front of obstacles.

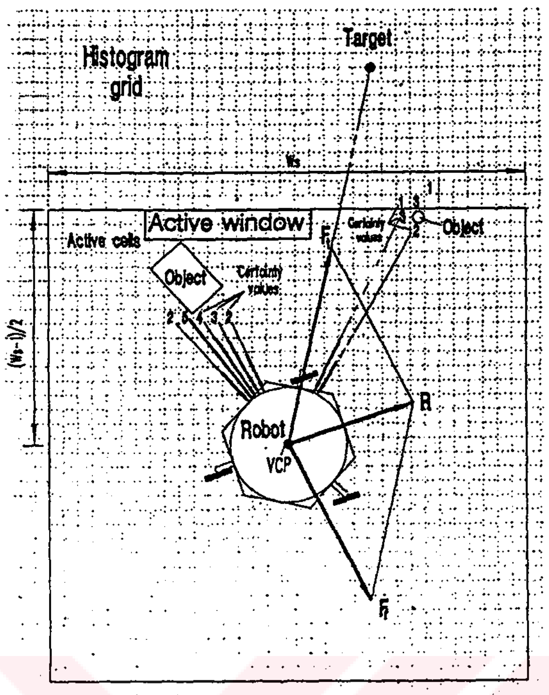


Figure 4.20 The Virtual Force Field Concept

The main idea in VFF is using a two-dimensional Cartesian grid called the histogram grid, c, for obstacle representation. Each cell (i,j), in the histogram grid holds a certainity value$(CV)_{i,j}$, that represents the confidence of the algorithm in the existence of an obstacle at that location. In the histogram grid, CVs are incremented when the range reading from a range-finder indicates the presence of an object at that cell. The Virtual Force Field can be seen in Fig 4.20. As the mobile robot moves, a square window accompanies it, overlying a region of C. this region is called Active region which is denoted as C^* and cells that momentarily belong to the Active Region are called Active Cells, which are denoted as $c^*_{i,j}$. In the implementation of Koren and Borenstein, the size of the windows is 33x33 cells and each cell has a size of 10cmx10cm, and the window is always centered about the robot's position. Each active cell exerts a virtual repulsive force $F_{i,j}$ toward the robot. The magnitude of this force is proportional to d^m, where d is the distance between the cell and the center of the vehicle, and m is a positive real number, which is usually 2.

$$\vec{F}(i,j) = \frac{F_{cr} * C(i,j)}{d^2(i,j)} \left(\frac{x_i - x_0}{d(i,j)} \vec{x} + \frac{y_i - y_0}{d(i,j)} \vec{y} \right) \qquad (4.2)$$

F_{cr}	: Repulsive Force constant,
d(i,j)	: Distance between cell (i,j) and mobile robot,
C(i,j)	: Certainity value of cell (i,j),
x_0, y_0	: The present coordinates of the mobile robot,
x_i, y_i	: The coordinates of cell (i,j),
$\vec{x}, \vec{y}$	: The vectoral representation of x and y

All virtual forces add up to yield the resultant repulsive force $\vec{F}_R$

$$\vec{F}_R = {}_{ij}\vec{F}(i,j) \qquad (4.3)$$

Simultaneously a virtual attractive force $\vec{F}_t$ of constant magnitude is applied to the mobile robot, pulling it toward the target.

$$\vec{F}_t = F_{ct} * \left(\frac{x_{t-}x_0}{d(t)} \vec{x} + \frac{y_{t-}y_0}{d(t)} \vec{y} \right) \qquad (4.4)$$

F_{ct}	: Repulsive force constant,
d(t)	: Distance between the target and the mobile robot,
x_t, y_t	: The coordinates of the target,
x_0, y_0	: The present coordinates of the mobile robot,
$\vec{x}, \vec{y}$	: Vectoral representation of x and y

Summation of $\vec{F}_R$ and $\vec{F}_t$ yields the resultant force vector $\vec{R}$.

$$\vec{R} = \vec{F}_R + \vec{F}_t \qquad (4.5)$$

$\vec{R}$: Resultant force vector

$\vec{F}_R$: Total attractive force vector.

$\vec{F}_t$: Total repulsive force vector.

The direction of the resultant force vector $\vec{R}$ is

$$\varphi = \text{atan}(\text{Im}(\vec{R}) / \text{Re}(\vec{R})) \qquad (4.6)$$

and the direction of $\vec{R}$ is used as the reference for the mobile robot's steering command.

One of the reason for the popularity of this method is its simplicity and elegance. It can be implemented quickly and initially provide acceptable results without require refinements. Also, no expensive precomputation as for C-space or Voronoi diagram is necessary. A smooth path is automatically created. As this approach uses only local information which could be obtained by sensors, it is fast enough to run in real-time. However the problem with this strictly local method is that the planner might get stuck in a local minimum and would not reach the final configuration.

CHAPTER 5. CONTROL STRUCTURES OF MOBILE ROBOTS

In order to achieve a high degree of robustness and autonomy, a mobile robot should be able to plan and execute actions and adapt its behavior to environment changes [61]. A mobile robot control structure should at least meet the following requirements:

1. **Programmability:** A useful mobile robot should be able to execute a variety of tasks without the need for changing its hardware nor software. To fulfill their task, a mobile robot must plan for their actions. The tasks should be given to the robot at an abstract level and the mobile robot should be able to interpret them adequately, according to its actual state and environment.
2. **Robustness:** The robot should not depend on a unique subsystem for all its actions, but be as redundant as possible in terms of sensors and functions.
3. **Autonomy:** Environment conditions can change, not be known perfectly and therefore a mobile robot should be able to handle its tasks itself, manage its various subsystems. This gives the robot autonomy.
4. **Adaptiveness:** It should be able to modify its behavior according to the actual conditions in it encounters. This feature requires reasoning capacities to detect situations and produce the adequate behavior.
5. **Reactivity:** It should be able to detect the events timely and react to them according to the context and the task. The reactivity should be adapted to the task. A mobile robot that is only driven by environment conditions may never achieve its goal because of possible behavior conflicts or systematic responses that may be not compatible with its overall objectives. Let's take for instance a mobile robot using proximity sensors which was designed to react to an obstacle by avoiding it. If this robot has to pick up an object on a table, it needs to close enough to the table. Proximity sensors will then trigger, leading the robot to avoid the table and preventing it from achieving the task. Therefore it is necessary that reactivity should be

programmable and controllable. According to the features mentioned above, the control structure of a mobile robot must have both some decision-making capacities and real-time capabilities.

There are essentially two forms of control architecture in use or in development at the present time. Firstly, there is the Functional Decomposition Fig. 5.1, in which the mobile robot is considered as a group of processes, known as agents, each carrying out a specific function.

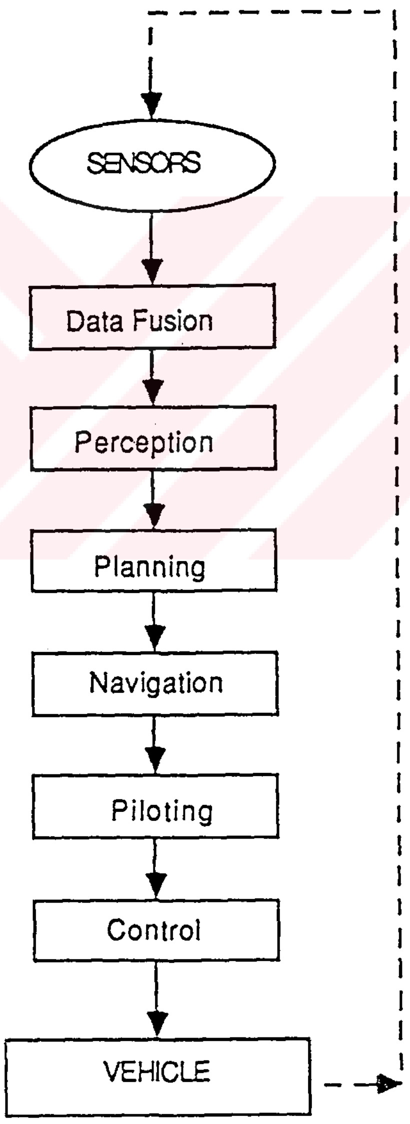


Figure 5.1 Functional decomposition of control structures

The agents are usually the perception system, the planner, the navigator and pilot, and the executor or controller. The second control structure is the Behavioral Decomposition, Fig. 5.2, which considers the mobile robot to be composed of levels of behavior such as avoid the obstacles, wander around, explore the environment, etc. In this structure, there is only performance degradation instead of total system failure when one of the higher levels fails. Additional layers of intelligent behavior must be added later, increasing the overall competence of the system.

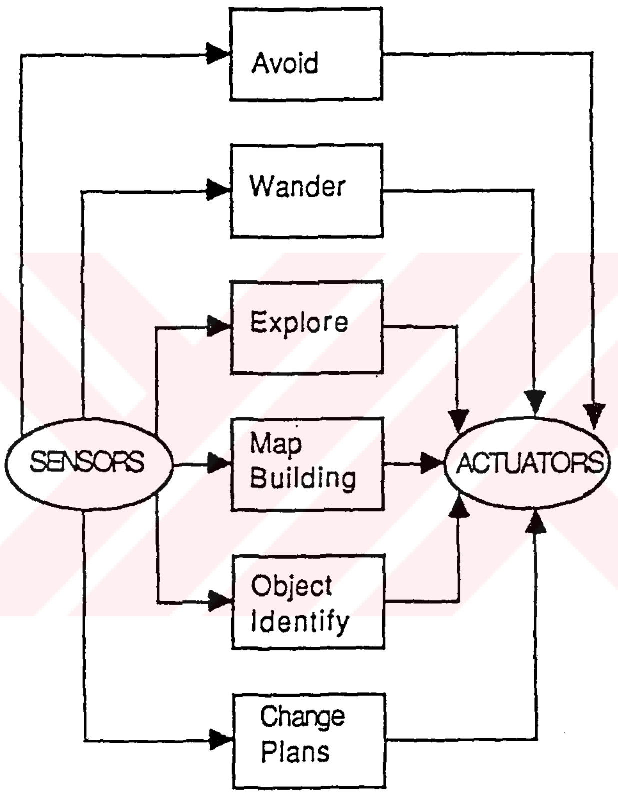


Figure 5.2 Behavioral decomposition of control structures

Both functional and behavioral control decompositions have their advantages and disadvantages. The behavioral decomposition and state transition layer are essential in the design of the control strategy, whereas the functional or hierarchical decomposition is important in the physical realization. In this chapter, functional and behaviorist decomposition control structures will be explained in details.

5.1. Functional Control Decomposition

5.1.1. Cognitive Control Structure for a Mobile Robot

Cognition is the main property of an autonomous mobile robot. Robotic cognition is the property and a process of perceiving and knowing; the system which makes this property operational. The process of rational activities, planning and decision making, image or concept recognition and/or criterion are traditionally assigned to the operation of the system of cognition [5]. Related to humanbeings, the term cognition presumes the phenomena of human mind. Robot cognition is expected to perform all the above mentioned operation based upon computers. According to this feature, the structure of a cognitive controller for a mobile robot is decomposed into the three levels of control.

1. Planner,
2. Navigator,
3. Pilot.

And it has the following four functional structures.

A. The mechanical assembly including the mobile platform, drive train, the subsystem of energy conversion and transmission, the actuator subsystem for acceleration, coasting and braking, the subsystem of steering, the subsystem of mounting and providing mechanical motion to the sensors.

B. The system of MOTION CONTROL, which is defined as a combination of principles, algorithms and develop destined to change the configuration of path, speed trajectory, the law of accelerating and decelerating, etc., in a way which will minimize the cost-function of the operation. Therefore the operations of motion control can be classified as:

a) Planning: Autonomous mobile robot control system will have to plan a set of motion operation to achieve the objectives. This will require the analysis of the goal location, analysis of terrain, analysis of constraints, preferences, expert information and for all the mechanisms their descriptive and mathematical models to identify a plan of operation It is a control of motion at the levels of strategic decision-making. The overall goal of the operation is synthesized on the basis of initial sensor information and the optimum trajectory of motion is determined and the rough trajectory of motion is sketched.

b) Navigation: It controls the motion at the tactile level. After the motion has started, the new information is perceived and the trajectory of motion is made more precise.

c) Piloting: It controls the motion at the operative level. This includes any vision-based decision-making processes performed in real-time on-line. Piloting implements the results of navigation. It transforms the prescribed trajectories into more specified and detailed sequence of maneuvers, including those not available for the algorithm of navigation since the resolution of the path, as well as the accuracy, is very high. Briefly, it is a conditional process and depends on the current state of the environment. As an example, the commanded speed of the mobile robot assembly must depend on actual surface condition, sudden turn, etc. Pilot control decision will have to be implemented that can accommodate this conditional process. Subproblems which depend on environment conditions and changes should be determined from the information of higher resolution and using the knowledge and reasoning power of the pilot level.

d) Execution Control: It is the lowest-level of control where the encoded solution is being transformed into the actual motion. Execution control, transforms the sequence of commands from the pilot into the actual motion. It is also a real-time control. it is based upon the need to follow the results of piloting as accurately as possible.

C. Perception: It is defined as system of sensors also. Perception includes all technological means of observing the real motion and judging on the closeness of the motion to the results of the prescribed and desired control, and also the technological means of proper transformation and integration of the measurement results. In otherwords perception system includes subsystem of sensors and subsystem of processing the information that is transmitted by sensors. It is used to monitor the motion control from the strategies, tactical and operative views. It provides the ability to extract information from the sensor data in order to construct an internal model of its environment. The perception subsystem uses a priori models of environment provided by the knowledge base and will output its world model into the knowledge base. Knowledge base is a system which performs automated extraction, storing, organization and generation of knowledge, as well as answering the external inquiries. The processing presumes dealing with knowledge stored in the system of knowledge and reasoning upon this information until the decision concerning the scene is obtained.

D. System of Knowledge: Knowledge is information which encompasses the present and the past experience of interest, and represented in a way which allows for interpretation, problem understanding, task generation and decision making. Knowledge hierarchy performs organizing the results of monitoring [62]. System of knowledge for the mechanical assembly is required in order to utilize properly the ideas implemented in the assembly by the design procedure. System of knowledge for the motion control system is required in order to determine the mobile robot tasks., to synthesize a solution concerned with the motion trajectories at all the considered levels of control. System of knowledge for the perception is also required in order to interpret the results of sensing and to properly assign the results of observations to the different levels of the hierarchy of motion control. The hierarchy of knowledge can be shown in Fig 5.3.

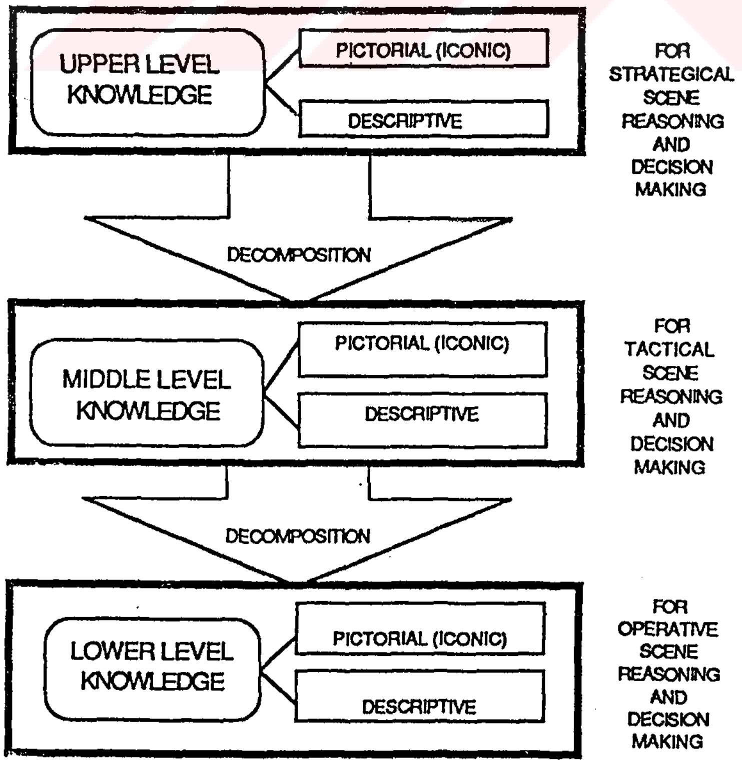


Figure 5.3 Hierarchy of knowledge

At each level of the hierarchy, the knowledge of the level is separated into the following 3 areas:

.1. Pictorial information which contains two groups of information : Visualized iconic images and transformed iconic images. Photoimages and topographical maps belong to this group.

2.World descriptive information which in turn is divided into two groups: Represented in natural or limited natural language and represented in symbolic language code.

3. Rules which include the collection of IF-THEN clauses organized in hierarchies according to the hierarchy of problems determined by the mobile robot mission. The hierarchy of IF-THEN clauses contain all rational knowledge of the world and of the mechanisms including the whole vehicle and the subsystem on-and off-board.

The relationship between the motion control system and the knowledge system are shown in Fig 5.4. If the output of each level can not be obtained and is inconsistent with reality, the upper level should reconsider its results and come down with a new solution. This can be called monitoring.

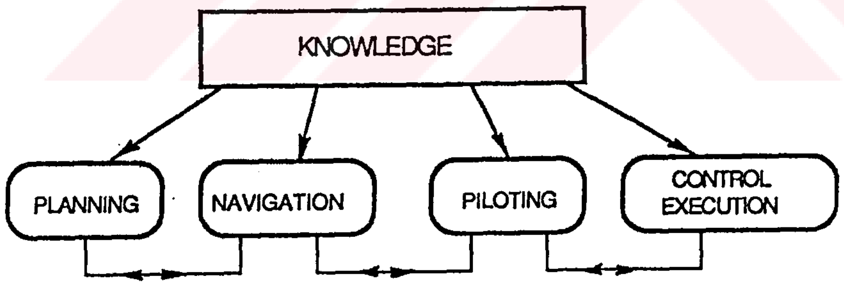


Figure 5.4 Relationships between knowledge and motion control

During execution of a plan, the system will be required to continually reevaluate its plan with regard to the currently inferred situation in order to determine if the process is succeeding or failing. When the process is failing, plan inadequacies must be identified and some degree of replanning should be initiated. During the trajectory execution, new information can determine the need in repeating the procedure of navigation. During the execution control operation, when an inability is revealed to follow the sequence of commands generated by PILOT, another maneuver must be assigned. The relationship between knowledge and vision system can be seen in Fig.5.5

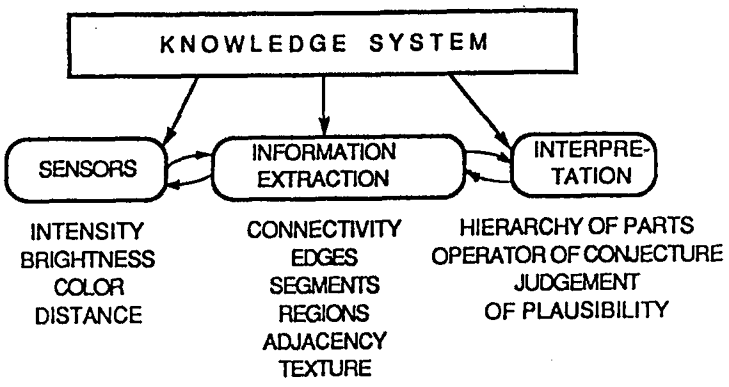


Figure 5.5 Relationships between knowledge and vision systems

In the system for processing information extraction, a judgment of the scene can be made when the output is transformed in some elements which help to make conjectures and try to verify them in the context. Therefore as it is shown in Fig 5.5, the connectivity of edges, segments, regions should be found and roughly organized in a matrix of adjacency. In information interpretation, a model of the environment to construct the observed scene using the tool of reasoning and making conjectures is made.

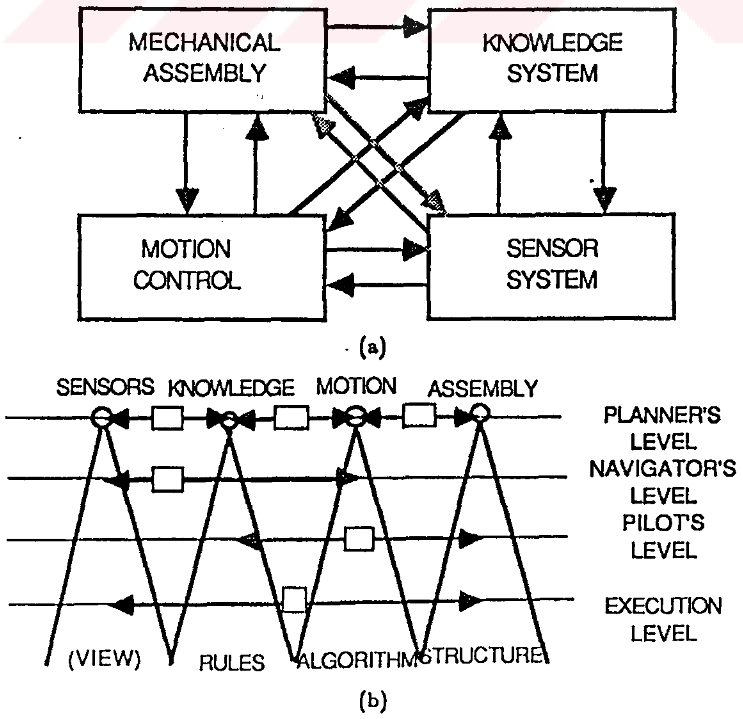


Figure 5.6 Functional and hierarchical structure of the cognitive controller

The interrelation between these four functional structures can be shown in Fig 5.6-a and their hierarchical character of the knowledge applied within these structures in Fig. 5.6-b. Interaction between the levels can be seen in Fig. 5.7. In cognitive control structure, the higher the level of hierarchy, the more is the time lag between the motion control determined and actually executed.

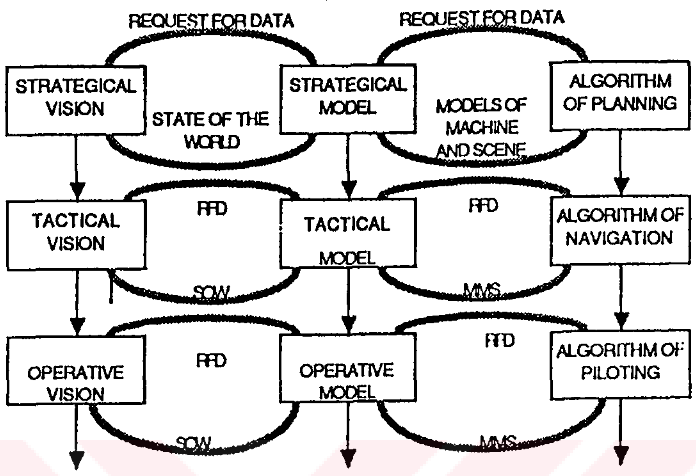


Figure 5.7 Interaction between the levels

5.1.2. The Distributed Control Architecture

The Distributed Control Architecture is a combination of expert modules which are synchronized by the aid of a blackboard. This is a centralized system because information is contained in a central database, modules must have low granularity due to the centralized communication and low communication bandwidth. In a software architecture for an autonomous mobile robot, several conceptual processing levels can be characterized. These processing levels are:

1. Robot Control: This level takes cares of the physical control of the different sensors and actuators available to the robot. It provides a set of primitives for locomotion, actuator and sensor control, data acquisition, etc., that serve as the robot interface, freeing the higher levels of the system from low level details. It includes activities such as vehicle-based motion, estimation and monitoring of internal sensors which provide information on the status of the different physical subsystems of the robot.

2. Sensor Interpretation: On this level, the acquisition of sensor data and its interpretation by sensor modules is done. Each sensor module is specialized in one type of sensor or even in extracting a specific kind of information from the sensor data. The modules provide information to the higher levels using a common representation and compatible frames of reference.

3. Sensor Integration: Due to the intrinsic limitations of any sensory device, it is essential to integrate information coming from qualitatively different sensors, such as stereo vision systems, sonar devices, laser range sensors, etc. Specific assertions provided by the sensor modules are correlated to each other on this level. For example, the geometric boundaries extracted from an obstacle detected by sonar can be used to provide connectivity information to a set of scattered three-dimensional points generated by the stereo vision subsystem. On this level, information is aggregated and assertions about specific portions of the environment can be made.

4. Real World Modeling: To achieve any substantial degree of autonomy, a robot system must have an understanding of its surroundings, by acquiring and manipulating a rich model of its environment of operation. This model is based on assertions integrated from the various sensors, and reflects the data obtained and the hypotheses proposed so far. On this level, local pieces of information are used in the incremental construction of a coherent global real-world model. This model can be used for several other activities, such as landmark recognition, matching of newly acquired information against previously stored maps, and generation of expectations and goals.

5. Navigation: For autonomous locomotion, a variety of problem-solving activities are necessary, such as local and global path planning, obstacle avoidance, detection of emergencies, etc. These different activities are performed by modules that provide specific services.

6. Control: This level is responsible for the scheduling of the different activities and for combining Data-Driven (responding to sensory information) and Plan-Driven (following a plan to execute a given task) activities in an integrated manner so as to achieve coherent behavior. Briefly, this level tries to execute the task-level plan that was handed to it, while adapting to changing real-world conditions as detected by the sensors.

7. Global Planning: To achieve a global goal proposed to the robot, this level provides task-level planning for autonomous generation of sequences of actuator,

sensor and processing actions. Other activities needed include simulation, error detection, diagnosis and recovery, and replanning in the case of unexpected situations or failures.

8. Supervisor: On this level a supervisory module oversees the various activities and provides an interface to a human user.

The communication between processing modules is not only possible between adjacent levels, but also there are a very complex interconnections and interdependencies between the various subsystems, with multiple flows of data and control. The general structure of the distributed control system which is proposed by Elfes [62] consists of an expandable community of Expert modules, which communicate asynchronously among themselves through messages. As it seen in Fig. 5.8 information needed for the common goal is posted on and retrieved from a Control Blackboard.

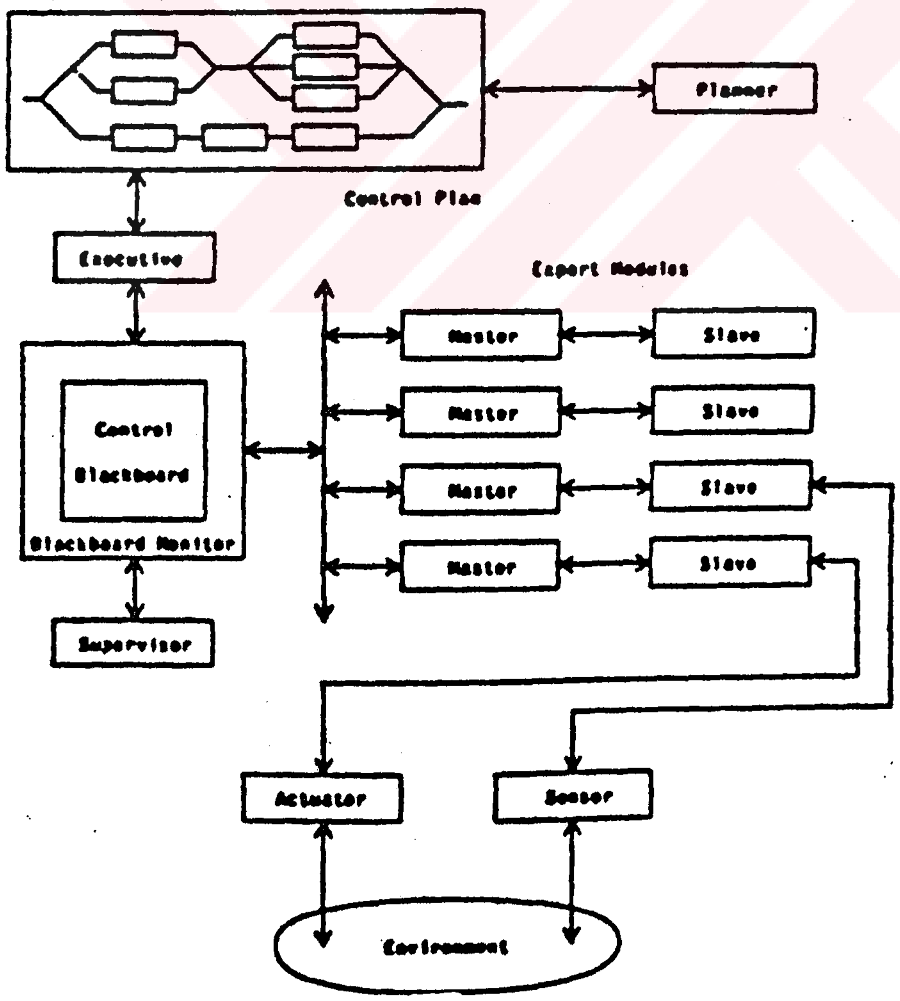


Figure 5.8 The distributed control structure

A Blackboard system is a problem-solving architecture [56]. It is described as a globally accessible database where different processes that are cooperating towards the solution of a certain problem can share data and results, suggest hypotheses to be examined, communicate the state of their own computations, etc.

In the control structure, a given task is performed under the direction of a Control Plan. Expert modules are specialized subsystems used to monitor the sensors and actuators, interpret sensory and feedback data, build an internal model of the mobile robot's environment, plan strategies to accomplish proposed tasks, supervise the execution of the plan and do global system management. Each expert module is composed of two closely coupled components: Master process and Slave process

Master process controls the scheduling and the activities of the Slave process and provides an interface to other modules. It retrieves needed data from the Blackboard, changes the status of the Slave and posts relevant information generated by the Slave process on the Blackboard.

Slave process is responsible for the processing and problem-solving activities. In Fig. 5.9, a module architecture is shown.

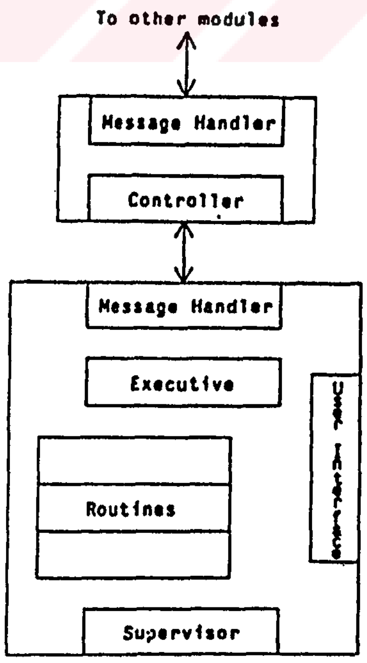


Figure 5.9 Expert module architecture

Each expert module has the following subsystems:

Message Handler: It handles incoming and outgoing traffic of information, either to other expert modules or to the Control Blackboard.
Controller: This is a knowledge-based subsystem that knows about the capabilities of its associates Slave. It invokes the Slave when appropriate, gathers the necessary data to provide to it, oversees the execution and changes the scheduling status of the Slave when necessary, and handles the results provided by it.
Module Executive: The module executive is responsible for the activities of the Slave. On the basis of the incoming requests and the data provided, it invokes the Internal Routines and performs all the processing required to execute the Slave's task. The results returned to the Master.
Module Supervisor: This subsystem allows the Slave module to run in stand-alone mode. It permits independent testing of the module, to facilitate debugging and integration.
User Interface: Through this subsystem, the user can inspect and change internal parameters of the module.

In the control structure, the Executor dynamically abstracts scheduling information for the Expert modules from the Control Plan. The Control Plan provides information specific to the execution of a given task by specifying subtasks and constraints in their execution. Over the Blackboard, high level information needed by the different subsystem is shared. This includes information on the robot's status, relevant interpreted sensory and feedback data and the scheduling information from the Control Plan. The Blackboard may be subdivided conceptually into subareas, where information specific to a certain problem or level of processing is stored. Access to the Blackboard is always handled by the Blackboard, and includes operations such as storing and retrieving information. The Monitor handles a queue of requests generated by other Knowledge Sources. This kind of encapsulation ensures uniformity of access to and consistency of updates on the Blackboard. The Knowledge Sources that operate on the Blackboard and interact with each other actually embody the knowledge that is being brought to bear on the solution to a given problem. Each one is typically dedicated to a particular type of subtask. Knowledge Sources that are working on a given problem suggest ways of decomposing a problem into simpler subproblems or favorable directions to explore in the search for a solution, or organize the sequencing of relevant task. The Expert modules are distributed over the processor network.

The control architecture that is proposed by Elfes reflects the structure of community of cooperating expert modules. These modules communicate asynchronously over the processor network, generating and absorbing streams of data. This structure embodies a hierarchical model of distributed computation. This shows the decision-making model. Control decisions are, whenever possible, made locally in the module which is confronted with a problem. Otherwise, the problem or data is broadcast recursively to the next higher level of decision making. Another result from this control structure is that commands and data can exist within the system at several levels of abstraction. At each level , only the necessary degree of detail is present. Higher levels are able to deal with the same information in a more abstract form and do not become cluttered with unnecessary details. The system described is loosely coupled since the rate of communication between modules, especially beyond the motor and sensor subsystem levels, is relatively small. This results from the use of asynchronous processes and the Blackboard. Such an approach lead to higher performance and better adaptability to dynamically changing conditions.

5.1.3. Hilare Control Architecture

The hierarchical control architecture used in Hilare mobile robot is organized into three main levels:

1. The lowest level is Functional Level. It is composed of a set of modules. A module performs a specific function that provides for a given capability of the robot related to perception and data processing, action and motion computation and closed loop processes. A module exchange data and requests with other modules and with the control level. Functional level is expandable namely modules may be added without modifying the global structure [63].

Modules in the functional level can be divided into three categories. These are:

a. Sensor/Effector Modules:

Sensor modules process data acquired by a physical sensor. The processings are generally organized into several levels corresponding to increasingly complex representations and abstract interpretations. representations resulting from one to an another level of processing are provided to other modules in the robot. Any level of representation within a sensor module may be used for event detection. For managing all the monitors within a sensor module, a Sensor Surveillance Manager is attached to each sensor module. Its role is to coherently achieve the condition tests, eliminating possibly redundant requests and to send a signal to the control system whenever the condition is met. The ultrasonic module of a mobile robot is an example of sensor module. Effector Modules drive actuator. There are different levels of command similar to the levels of interpretation in the sensor modules. The Pilot module, designed to drive the robot's motorized wheels, is an example of an effector module and it provides for trajectory control and tracking.

b. Servo-process Modules :

Servo-process modules provide for a dynamic link between perception and action. They are used to servo robot motion using sensory data. a servo-process module has three types of links. Input lines carrying data from sensor modules; output lines to issue commands to effector modules or send data to another module; control line from and towards the control system. The output line can be routed to be the input of different modules. The control line is used to send activation of halt signals to the servo-process and to retrieve a failure signal from it when the input data do not enable it to run correctly. When a servo-process request data from a sensor module hat can not provide it, the servo-process reports to the control system. Obstacle avoidance module, wall following module and visual tracking module are examples of servo-process modules. Surveillance Monitor play a very important role in the activation and inhibitation of the sensor processes.

c. Functional Units:

The information provided by sensor modules at several level of interpretation may be used by dedicated modules which are called Functional Units. These modules are activated upon a request to execute some computations, possibly making use of other modules. Trajectory planning and mobile robot localization are example of functional units.

2. The intermediate level is Control System Level translates plans given by the planner level into tasks and request to the modules, controls the robot's actions to execute the task and manages its resources. It also sets robot modules to detect events and provides for correcting actions and error recovery.

The Control System organizes the robot's motion in order to achieve a consistent behavior, to execute plans and reports information when failures occur [64]. It is responsible for adapting the robot's behavior to ensure reactivity to new situations. The Control System has four components as it is shown in Fig 5.10.

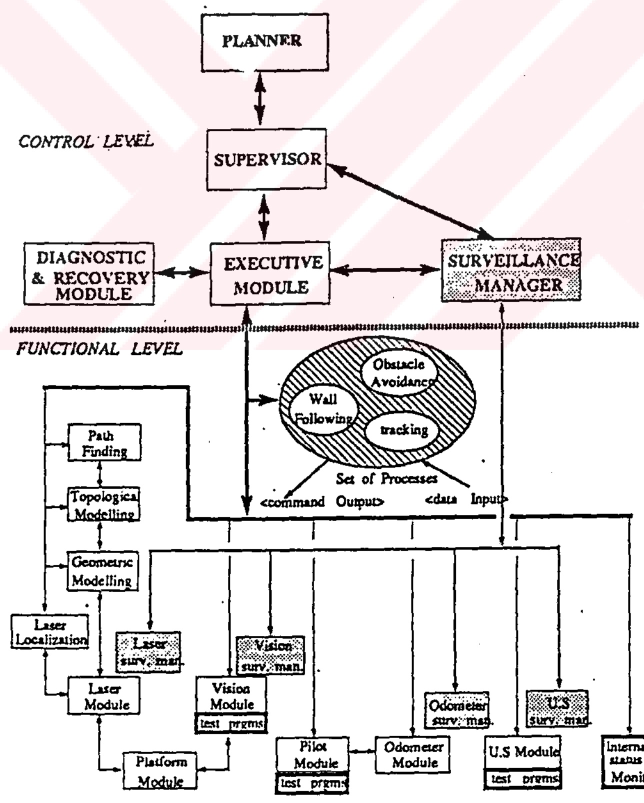


Figure 5.10 The Control architecture of Hilare robot

a) Supervisor Module: The main goal of the Supervisor Module is to receive a plan and control its execution as a whole and to take into account events that might have a global influence on its achievement. Each step of a plan corresponds to a task which is sent to the Executive to be parsed and the fulfilled. The Supervisor may also receive dynamically tasks produced by other modules in the architecture. The supervisor interacts with the Planner. In order to specify tasks between Supervisor and the Executive levels, a task description language is designed.

b) Surveillance Manager: The robot reacts to an event in three stages: First by a reflex reaction, second by an action determined by the current task and third by task replanning. To provide this programmable reactivity, Surveillance Monitors are introduced. Surveillance Monitors play a key role in mission execution and in robot operation. Their purpose is to set condition on sensed data or computed results and to trigger some reflex reaction when the conditions are met. Two mains of Surveillance monitors in the control system are defined. P-Surveillances and Q-Surveillances. P-Surveillances related to the on-going plan or mission, and provided along with it to detect some events or situation and cope with them. Q-Surveillances related to the operation of the robot, even if no task is being executed.

All monitors are set by the Surveillance Manager that is informed when they are fired. However for each sensor module, there is also a local surveillance manager. Conditions can be set at any level of data processing in a chain of sensor modules as their results in order to guarantee an immediate reaction when the events that produced them occur. On the same sensor module, several surveillance monitors can be active at the same time and are controlled by a Sensor Surveillance Manager. At the top level of the Surveillance System is the Surveillance Manager. Its functions are:

- It receives surveillance monitors from the Executive and decomposes the condition part, sending the various components to the adequate Sensor Surveillance Managers.
- It receives the messages from the SSMs indicating that a surveillance has been triggered. I f the condition part of a given monitor included a logical condition on different sensors, the SM verifies the logical formula. Then the reflex actions corresponding to the triggered monitors are executed through the Executive. If different surveillance monitors are triggered simultaneously, a priority order is used to decide what action to perform first.

c) Executive Module: The Executive carries out tasks, manages robot resources and detect conflicts. In a way, this module is similar to an operating system of the robot. a task is represented as a succession of command written in task description language. If a command is executed correctly, the Executive launches the next one. Otherwise, in case of failure, the Executive calls the Diagnostic and Error Recovery Module to try to achieve the task.

d) Diagnostic and Error Recovery Module: In an error situation, The Executive perceives a flow of errors from the modules engaged in the execution of a task. After stopping the execution, the flow is transmitted to the Diagnostic and Error Recovery module. For each command that was executed, Diagnostic and Error Recovery module first builds a failure three deduced from the execution status provided by tokens and the error messages. The root of this tree contains the fault statement and three attributes are associated with it. "Error" is the list of the error messages issued by the module directly responsible for execution of the statement, "Contain" lists all the other modules invoked by module and "May-contain" lists the modules that may possibly be invoked by module. The next stage of the analysis is to find out the primitive error responsible for the list of errors. This is very complex, if there are interdependency between faults. Once the fault has been found, the action to take so as to recover the failed statement is given by other prestored information. The task may be aborted or mended by adding a list of statements to be executed. Mending may also consist of substituting an order by another one. For example, a servo-process using vision to follow a corridor may be substituted by another one using ultrasonic sensors.

3. The highest level is the Planning Level. It generates plans to achieve goals. Decomposition of the system into a Planning Level and Control Level is related to hierarchical planning. Since the planning level uses abstract, partial, and uncertain world models, it must leave as much latitude as possible to the Control Level to adapt to the actual state of the world, while not being ambigious. To achieve this, the planning level must provide an executable plan to ensure nonambigious interpretation and robust execution.

This control structure uses the concept of process to achieve a controlled reactivity through the use of surveillance monitors providing a flexible control of these processes. It has reactivity and error recovery capacities and a fixed hierarchical structure.

5.2. Behavioral Control Decomposition

5.2.1. Layered Control Architecture

As an behaviorist control striker, Brooks [65] has proposed a number of levels of competence for an autonomous mobile robot. A level of competence is an informal specification of a desired class of behaviors for a mobile robot over all environments it will encounter. A higher level of competence implies a more specific desired class of behaviors. As an application example for the levels of competence for a mobile robot can be given as follows:

0. Avoid contact with objects whether the objects move or stationary.
1. Wander aimlessly around without hitting things.
2. Explore the world by seeing places in a distance that reachable and heading for them.
3. Build a map of the environment and plan routes from one place to another.
4. Notice changes in the static environment.
5. Reason about the world in terms of identifiable objects and perform task related to certain objects.
6. Formulate and execute plans that involve changing the state of the world in some desirable way.
7. Reason about the behavior of objects in the world and modify plans accordingly.

It should be noticed that each level of competence includes as a subset each earlier level of competence. Since a level of competence defines a class of valid behaviors, it can be seen that higher levels of competence provide additional constraints on that class. The most important idea of levels of competence is, layers of a control system corresponding to each level of competence can be built and simply added a new layer to an existing set to move to the next higher level of overall competence. Control is layered with higher level layers subsuming the roles of lower level layers when they wish to take control. The system can be partitioned at any level and the layers below form a complete operational control system as it is seen in Fig 5.11. This architecture is called Subsumption Architecture.

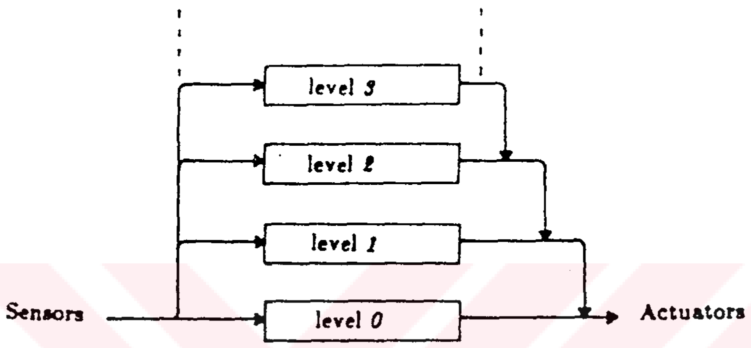


Figure 5.11. Subsumption architecture

In a Subsumption architecture, firstly Level Zero competence is built. It is called also Zeroth -Level control system. Next the first-level control system is built. First-level control system is able to examine date from the level zero system and is also permitted to inject data into the internal interfaces of level zero suppressing the normal data flow. This layer, with the aid of the zeroth, achieves level one competence. The zeroth layer continues to run unaware of the layer above it which sometimes interferes with its data paths. After the first layer is built, additional layers can be added and the initial working system need never be changed.

Layers are built from a set of small processors that send messages to each other. Each processor is a finite state machine with the ability to hold some Lisp data structures. Processors send messages over connecting wires. There is no handshaking or acknowledgment of messages. The processors run completely asynchronously, monitoring the input wires and sending messages on their input wires. It is possible for messages to get lost. There is no shared global memory so there is no other form of communication between processors. All modules are created equal in the sense that within a layer there is no central control. Each module does its task as best it can. Inputs to modules can be suppressed and outputs can be inhibited by wires terminating from other modules. This is the mechanism by which higher level layers subsume the role of lower levels. The control structure can be defined in more details by a mobile robot control system which achieves level zero and level one competence as defined before.

A. Zeroth Level: The lowest layer of control makes sure that the robot does not contact with other objects. In Fig 5.12 level zero competence is shown and given a complete description of how modules are connected together.

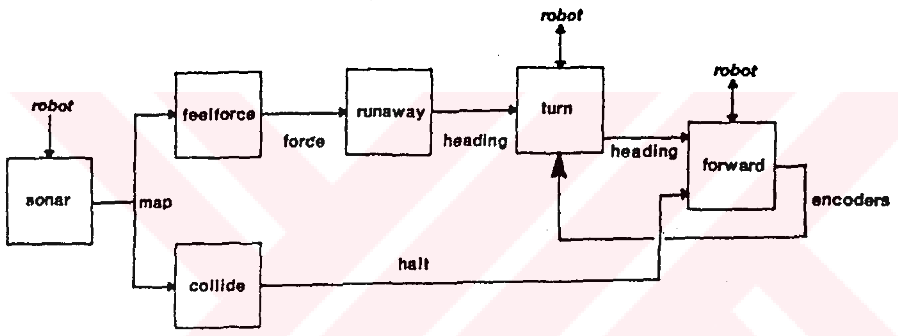


Figure 5.12 Level 0 control system

If an object approaches the robot, it will move away. If in the course of moving itself it is about to collide with an object, it will halt. These actions are sufficient for the level zero competence. The Turn and Forward modules communicate with the actual robot. They have extra communication mechanism, allowing them to send and receive commands to and from the physical robot directly. The Turn module receives a heading specifying an in-place turn angle followed by a forward motion of a specified magnitude. It commands the robot to turn and on completion passes on the heading to the Forward module. The Turn module then goes into a wait state ignoring all incoming messages. The Forward module commands the robot to move forward but stops the robot if it receives a message on its halt input line during motion. as soon as the robot is idle, it sense out the shaft encoder readings. The message acts as a reset for the Turn module, which is then once again ready to accept a new motion command. The Sonar module takes a vector of sonar readings, filters them for invalid readings, and effectively produces a robot centered map of obstacles in polar coordinates. The Collide module monitors the sonar map and if it detects objects dead ahead, it sends a signal on the halt line to the Motor module. The Collide module does not know or care if the robot is moving or not. Halt messages sent while the robot is stationary are essentially lost. The Feelforce module sums the results of considering each detected object as a repulsive force, generating a single resultant force. The Runaway module monitors the force produced by the sonar detected obstacles and sends commands to the turn module if it ever becomes significant.

B. First level: The first level layer of control, when combined with the zeroth, imbues the robot with the ability to wander around aimlessly without hitting objects. It should be noticed that this control level relies in a large degree on the zeroth level's aversion to hitting obstacles.

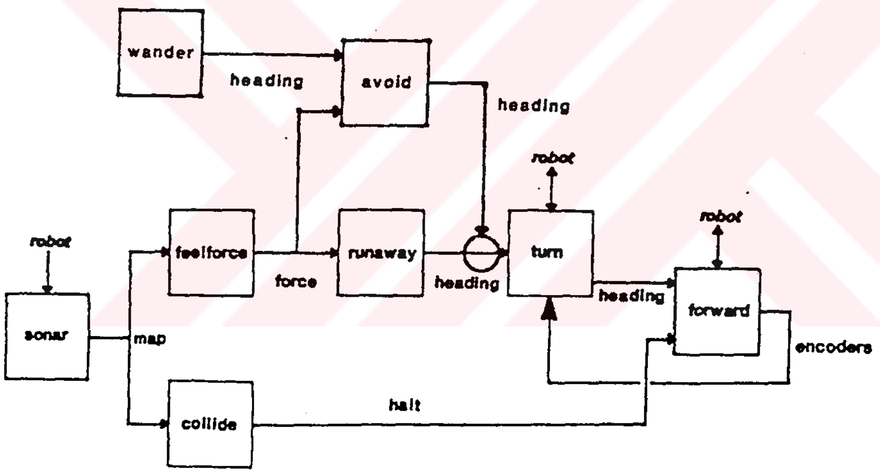


Figure 5.13. Level 0 system augmented with the level 1 system

In addition it uses a simple heuristic to plan ahead a little in order to avid potential collisions which would need to be handled by the zeroth level. The Wander module generates a new heading for the robot in every specified period. The Avoid module takes the result of the force computation from the zeroth level and combines it with the desired heading to produce a modified heading, which usually points in roughly the right direction, but is perturbed to avoid any obvious obstacles. This computation implicitly subsumes the computations of the Runaway module, in the case that there is also a heading to consider. In fact the output of the Avoid module suppresses the output from the Runaway module as it enters the Motor module. In Fig. 5.13 a complete description of how the modules are connected together is given. It should be noticed that the first level is simply level zero control system with some more modules and wires added.

The characteristics of the robust layered control system are as follows:

- The mobile robot control problem can be decomposed in terms of behaviors rather than in terms of functional modules. The interaction between behaviors ensures reactivity. A lower level can be subsumed by a higher level, inhibiting the input/output of the lower layer.
- It provides a way to incrementally build and test a complex mobile robot control system. Lower levels that have been well debugged continue to run when higher levels are added. Since a higher level can only suppress the outputs of lower levels by actively interfering with replacement data, in the case that it can not produce results on time, the lower levels still produce sensible results.
- One of the motivations for developing the layered control system is extensibility of processing power. Useful parallel computation can be performed on a low bandwidth loosely coupled network of asynchronous simple processors. The topology of that network is relatively fixed.
- There is no need for central control module of a mobile robot. The control system can be viewed as a system of agents each busy with their individual goal. There is no need to make an early decision on which goal should be pursued.

Briefly, this methodology decomposes a control system into a set of loosely coupled task achieving behaviors. Each behavior is a complete control system going from sensory inputs to motor outputs. Once a behavior is debugged, it is no more changed. Sophisticated control systems can be build around it because of it is simple, extensible arbitration scheme. The main drawback of the robust layered control system that it is entirely wired, so it is permanent and irrevocable. The robot is also not programmable and is limited to the behaviors that were implemented by its designer.

CHAPTER 6. PTP MOTION CONTROL OF A MOBILE ROBOT

6.1. The Mobile Robot Kinematics

The mobile robot which is considered in this study has a three-wheeled configuration. There are two wheels on front and one wheel on the back of the robot. Each one of the front wheels are connected by an independently controlled DC motor. The back wheel is a caster wheel and it can turn to every directions. This type of steering is called as differential steering because steering is effected by differences in the velocity of the two front wheels. An advantage of differentially driven mobile robot is the ability to turn on a dime, that is they have a zero turning radius.

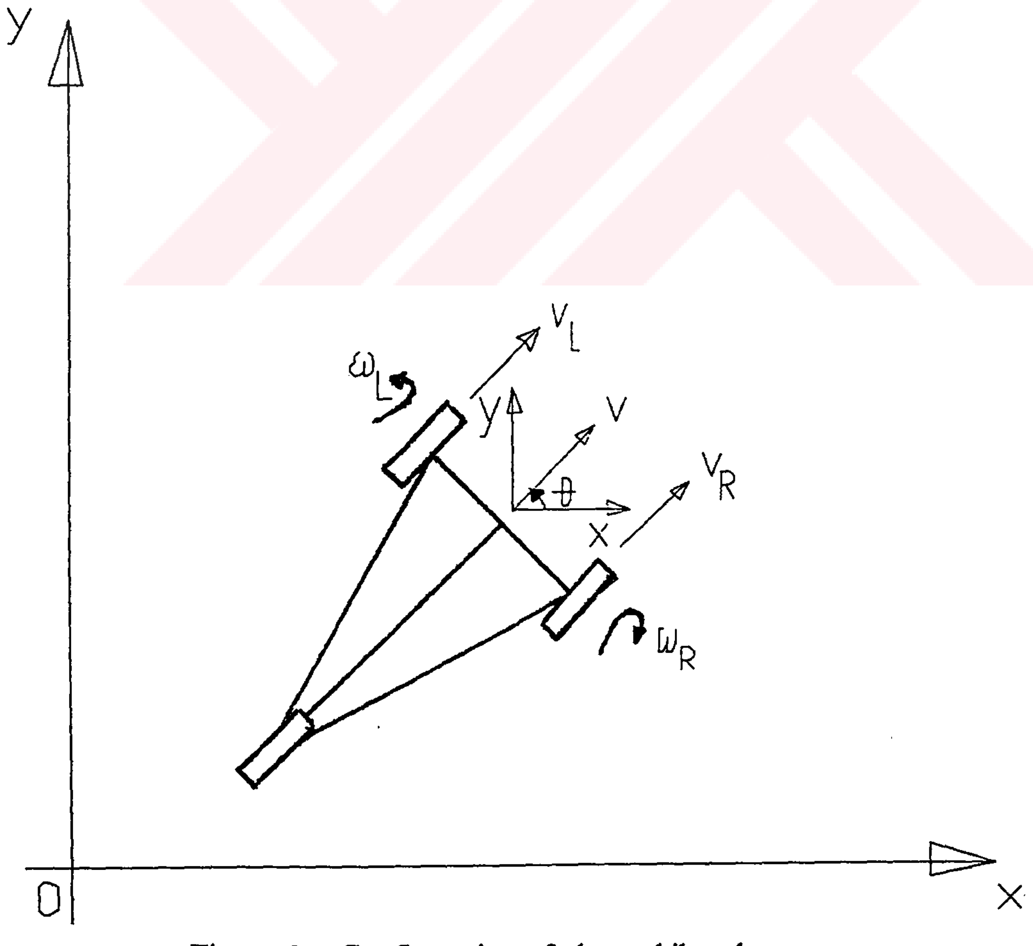


Figure 6.1 Configuration of the mobile robot

The vehicle's position and orientation is represented by three parameters (x,y,θ), two for translation and one for orientation, when only planar motion is assumed. However, for a common tricycle and differential-drive mobile robot configurations, there are only two degrees of freedom for control, that is, direction angle and velocity or the independent velocity of the two wheels of a differential-drive vehicle. Such systems are referred to as nonholonomic. The kinematic equations of the vehicle can be written as follows:

The right wheel linear velocity:

$$V_R = r\omega_{w_R} \tag{6.1}$$

The left wheel linear velocity:

$$V_L = r\omega_{w_L} \tag{6.2}$$

The center of mass linear velocity:

$$V = \frac{V_R + V_L}{2} = \frac{r}{2}(\omega_{w_R} + \omega_{w_L}) \tag{6.3}$$

where, r is the radius of the front wheels of the mobile robot. To find an equation for the direction angle:

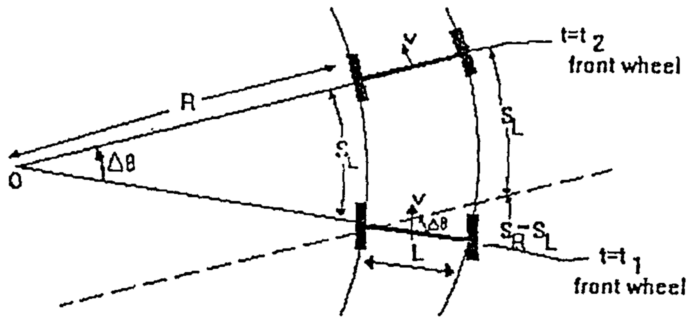


Figure 2 Angle from instantaneous motion

where,

O:	Instantaneous center of motion
S_R, S_L:	Displacement of the right wheel and left wheel, respectively

Δt is time interval and,

$$\Delta t = t_2 - t_1 \tag{6.4}$$

The length of the arcs are obtained as follows:

$$S_R = \Delta t V_R \tag{6.5}$$

$$S_L = \Delta t V_L \tag{6.6}$$

$$\Delta\theta L = S_R - S_L \tag{6.7}$$

$$\Delta\theta L = \Delta t(V_R - V_L) = \Delta t(\omega_{w_R} - \omega_{w_L})r \tag{6.8}$$

When the both side of the equation (6.8) is divided by Δt, equation (6.9) will be obtained as follows :

$$\frac{\Delta\theta}{\Delta t} = \frac{r}{L}(\omega_{w_R} - \omega_{w_R}) \quad \Rightarrow \frac{d\theta}{dt} = (\omega_{w_R} - \omega_{w_L}) \tag{6.9}$$

According to the direction angle of the vehicle, x component of the linear velocity V_x and y component of the linear velocity V_y are obtained in equation (6.10) and equation (6.11), respectively.

$$\frac{dx}{dt} = V_x = V\cos\theta = \frac{r}{2}(\omega_{w_R} + \omega_{w_L})\cos\theta \tag{6.10}$$

$$\frac{dy}{dt} = V_y = V\sin\theta = \frac{r}{2}(\omega_{w_R} + \omega_{w_L})\sin\theta \qquad (6.11)$$

x and y positions of the mobile robot can be calculated, by taking the integrals of the equations (6.10) and (6.11), respectively.

6.2. Dynamics of the Mobile Robot

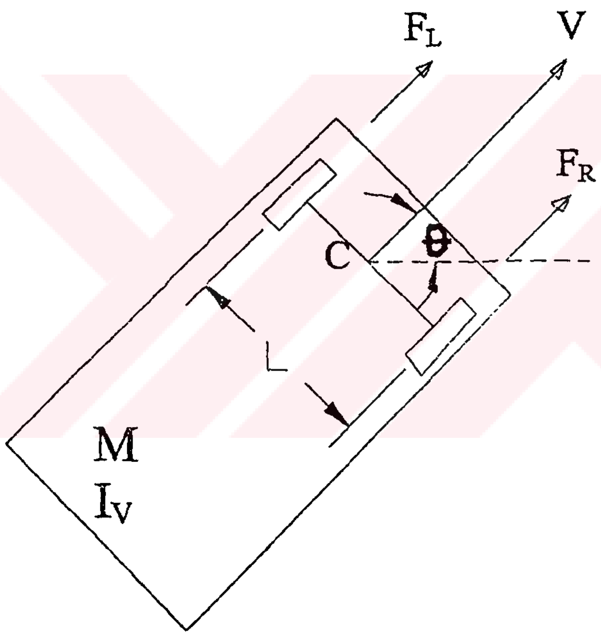


Figure 6.3 Dynamics of the vehicle

As it is shown in Fig. 6.2, the right wheel traction force and left wheel traction force of the vehicle are F_R and F_L, respectively. When the rotational motion of the vehicle is considered, equation (6.12) and when the translational motion of the vehicle is considered, equation (6.13) are obtained respectively.

$$(F_R - F_L)\frac{L}{2} = I_v \ddot{\theta} \qquad (6.12)$$

$$F_R + F_L = M\dot{V} \qquad (6.13)$$

The symbols used in equation (6.12) and equation (6.13) are given below:

M : Mass of the vehicle.

I_v : Yaw moment of inertia of the vehicle

L : Linear distance between front wheels.

V: Linear velocity of the vehicle.

θ : Direction angle (orientation) of the vehicle.

In Fig. 6.3., the driving mechanism of one of the front wheels is shown and the other front wheel is driven similar and has the same characteristics. The torque to the wheel is provided by a separately excited DC motor, through a gear box.

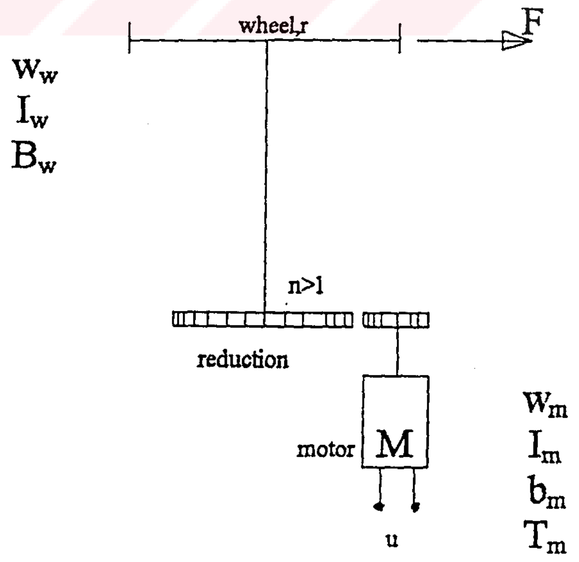


Figure 6.4 The driving mechanism of one of the front wheels

The symbols used in Figure 6.4 are given below:

T_m	:	Torque developed by motor
I_m	:	Moment of inertia of the motors of the mobile robot
b_m	:	Viscous friction coefficient of the motors of the mobile robot
u	:	Voltage enters the motor
n	:	Reduction ratio (n>1)
ω_w	:	Angular velocity of the wheel
ω_m	:	Angular velocity of the motor
r	:	Radius of the wheels of the mobile robot
I_w	:	Moment of inertia of the wheels of the mobile robot
B_w	:	Viscous friction coefficient of the wheels of the mobile robot
F	:	Traction force of the wheel

Considering dynamics of the right front wheel driving mechanism :

$$\left(T_{m_R} - I_{m_R}\dot{\omega}_{m_R} - b_{m_R}\omega_{m_R}\right)n = I_{w_R}\dot{\omega}_{w_R} + B_{w_R}\omega_{w_R} + F_R r \qquad (6.14)$$

equation (6.14) is obtained.

The angular velocity of the right wheel is obtained as follows:

$$\omega_{w_R} = \frac{\omega_{m_R}}{n} \qquad (6.15)$$

The DC motor dynamics for the right wheel is given in equation (6.16)

$$T_R = k_1 u_R - k_2 \omega_{m_R} \qquad (6.16)$$

where k_1 is motor-voltage constant and k_2 is the back e.m.f. voltage constant.

The right and left traction forces can be obtained from equation (6.1) and equation (6.2) as follows:

$$F_R = \left(\frac{M\dot{V}}{2} + \frac{I_v\ddot{\theta}}{L}\right) \qquad (6.17)$$

$$F_L = \left(\frac{M\dot{V}}{2} - \frac{I_v\ddot{\theta}}{L}\right) \qquad (6.18)$$

When equation (6.15), equation (6.16), are substituted into equation (6.14), the following equation (6.19) is obtained.

$$(k_1 u_R - k_2 \omega_{w_R} n - I_{m_R} \dot{\omega}_{w_R} n - b_{m_R} \omega_{w_R} n)n = I_{w_R} \dot{\omega}_{w_R} + B_{w_R} \omega_{w_R} + F_R r \qquad (6.19)$$

When equation (6.17) is substituted into equation (6.19) and equation (6.19) is rearranged, equation (6.20) is obtained for the right wheel driving mechanism.

$$k_1 u_R n - k_2 \omega_{w_R} n^2 - (I_{w_R} + I_{m_R} n^2)\dot{\omega}_{w_R} - (B_{w_R} + b_{m_R} n^2)\omega_{w_R} = \left(\frac{M\dot{V}}{2} + \frac{I_v\ddot{\theta}}{L}\right) r$$

(6.20)

When the same operation is applied for the left driving mechanism, equation(6.21) is obtained.

$$k_1 u_L n - k_2 \omega_{w_L} n^2 - (I_{w_L} + I_{m_L} n^2)\dot{\omega}_{w_L} - (B_{w_L} + b_{m_L} n^2)\omega_{w_L} = \left(\frac{M\dot{V}}{2} - \frac{I_v\ddot{\theta}}{L}\right) r$$

(6.21)

When both sides of equation (6.20) and equation (6.21) are summed:

$$k_1 n(u_R + u_L) - k_2 n^2(\omega_{w_R} + \omega_{w_L}) - (I_w + I_m n^2)(\dot{\omega}_{w_R} + \dot{\omega}_{w_L}) - (B_w + b_m n^2)(\omega_{w_R} + \omega_{w_L})$$
$$= M\dot{V}r$$

(6.22)

Equation (6.22) is obtained. When equation (6.22) is rearranged according to equation (6.3),

$$\left(Mr + \frac{2I_w}{r} + \frac{2I_m n^2}{r}\right)\dot{V} + \left(\frac{2k_2 n^2}{r} + \frac{2B_w}{r} + \frac{2b_m n^2}{r}\right)V = k_1 n(u_R + u_L)$$

(6.23)

is obtained.

When equation (6.21) is subtracted from equation (6.20),

$$k_1 n(u_R - u_L) - k_2 n^2(\omega_{w_R} - \omega_{w_L}) - (I_w + I_m n^2)(\dot{\omega}_{w_R} - \dot{\omega}_{w_L}) - (B_w + b_m n^2)(\omega_{w_R} - \omega_{w_L})$$
$$= \frac{2I_v\ddot{\theta}}{L} r$$

(6.24)

equation (6.24) is obtained.

When equation (6.24) is arranged according to equation (6.9),

$$\left(\frac{2I_v r}{L} + \frac{I_w L}{r} + \frac{I_m n^2 L}{r}\right)\ddot{\theta} + \left(\frac{B_w L}{r} + \frac{b_m n^2 L}{r} + \frac{k_2 n^2 L}{r}\right)\dot{\theta} = k_1 n(u_R - u_L)$$

(6.25)

is obtained.

As a result, equation (6.23) and equation (6.25) gives the dynamics of the vehicle.

6.3. State Space Model of the Mobile Robot

For the ease of arranging in advance, dynamic equations of the vehicle, equations (6.23) and (6.25) can be written as follows:

$$A\dot{V} + BV = C(u_R + u_L) \tag{6.26}$$

$$D\ddot{\theta} + E\theta = C(u_R - u_L) \tag{6.27}$$

where,

$$A = \left(Mr + \frac{2I_w}{r} + \frac{2I_m n^2}{r} \right)$$

$$B = \left(\frac{2k_2 n^2}{r} + \frac{2B_w}{r} + \frac{2b_m n^2}{r} \right)$$

$$C = k_1 n \tag{6.28}$$

$$D = \left(\frac{2I_v r}{d} + \frac{I_w L}{r} + \frac{I_m n^2 L}{r} \right)$$

$$E = \left(\frac{B_w L}{r} + \frac{b_m n^2}{r} + \frac{k_2 n^2 L}{r} \right)$$

According to equation (6.26) and equation (6.27), as state variables of the system,

$$x_1 = V, \qquad x_2 = \dot{\theta}, \qquad x_3 = \theta \tag{6.29}$$

are selected. From equations (6.26) and (6.27) , the states can be written as follows:

$$\dot{x}_1 = -\frac{B}{A}x_1 + \frac{C}{A}(u_R + u_L) \qquad (6.30)$$

$$\dot{x}_2 = -\frac{E}{D}x_2 + \frac{C}{D}(u_R - u_L) \qquad (6.31)$$

$$\dot{x}_3 = x_2 \qquad (6.32)$$

The equations (6.30), (6.31), and (6.32) can be shown also in matrix form as follows:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \end{bmatrix} = \begin{bmatrix} -B/A & 0 & 0 \\ 0 & -E/D & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \begin{bmatrix} C/A & C/A \\ C/D & -C/D \\ 0 & 0 \end{bmatrix} \begin{bmatrix} u_R \\ u_L \end{bmatrix} \qquad (6.33)$$

Briefly, the system can be written in the form:

$$\dot{x} = Ax + Bu \qquad (6.34)$$

where,

$$x = \begin{bmatrix} x_1 & x_2 & x_3 \end{bmatrix}^T \qquad (6.35)$$

$$u = \begin{bmatrix} u_R & u_L \end{bmatrix}^T \qquad (6.36)$$

and,

$$A = \begin{bmatrix} -B/A & 0 & 0 \\ 0 & -E/D & 0 \\ 0 & 1 & 0 \end{bmatrix} \qquad (6.37)$$

and,

$$B = \begin{bmatrix} C/A & C/A \\ C/D & -C/D \\ 0 & 0 \end{bmatrix} \qquad (6.38)$$

6.4. Control of the Mobile Robot:

The motion control algorithm of the mobile robot is point-to-point (PTA) method. Motion control means the strategy by which the vehicle approaches a desired location and implementation of this strategy. The PTP method is used to control the vehicle to follow given target points (x_{t_i}, y_{t_i}) where , i=1,2,3................n. In order to reach the target points, linear velocity and direction angle of the vehicle are controlled.

For linear velocity control of the vehicle, PI controller is used. PI controller performs the operation of amplification (P=Proportional) and integration (I=Integral) on the error signal fed into the controller as the input. The P-part influences the steady-state gain and I-part influences the steady-state error. Each part of the controller has gains to be adjusted or tuned experimentally so that the desirable responses of the system are obtained.

The PI controller input is the error between reference velocity V_{ref} and actual velocity V of the mobile robot. In our plant, V is represented by x_1 as state variable. The PI controller output u_1 is obtained as follows:

$$u_1 = K_{p_v}(V_{ref} - x_1) + K_{i_v}\int_0^t (V_{ref} - x_1)dt \qquad (6.39)$$

K_{p_v}	: Proportional gain for linear velocity controller
$K_{i_v} = \dfrac{K_{p_v}}{\tau_i}$	: Integral gain for linear velocity controller

where τ_i is a constant called integral time.

(x_{t_i}, y_{t_i}) which is the position of the (i)th target point is assumed as the start position of the vehicle and $(x_{t_{i+1}}, y_{t_{i+1}})$ is the position of the (i+1)th target point (i=1,2,3, ...,n). According to the distance between two consecutive target points, reference velocity V_{ref} is obtained as follows:

$$V_{ref} = \frac{\sqrt{(x_{t_{i+1}} - x_{t_i})^2 + (y_{t_{i+1}} - y_{t_i})^2}}{\Delta T} \qquad (6.40)$$

ΔT is the unit time that will take the mobile robot to reach the (i+1)th target point. The velocity reference is defined in every unit time ΔT as (with a maximum V_{ref} m/sec) trapezoidal shape.

For direction angle (orientation) control of the vehicle, PD controller is used. PD controller performs the operation of amplification (P=Proportional) and differentiation (D=Derivative) on the error signal fed into the controller as the input. The P-part influences the steady-state gain and D-part speeds up the response by performing a predictive-type function. Each part of the controller has gains to be adjusted or tuned experimentally so that the desirable responses of the system are obtained.

The PD controller input is the error between reference direction angle θ_{ref} and actual direction angle θ of the mobile robot. In our plant, θ which is the direction angle of the vehicle is represented by x_3 and $\dot{\theta}$ which is the angular velocity of the vehicle is represented by x_2 as state variables. The PD controller output u_2 is obtained as follows:

$$u_2 = K_{p_\theta}(\theta_{ref} - x_3) + K_{v_\theta}(w_{ref} - x_2) \qquad (6.41)$$

Since angular velocity reference w_{ref} of the vehicle is desired to be zero and the derivative of x_3 is x_2.

$$u_2 = K_{p_\theta}(\theta_{ref} - x_3) - K_{v_\theta} x_2 \qquad (6.42)$$

is obtained.

K_{p_θ}	:	Proportional gain for direction angle controller
$K_{v_\theta} = \tau_d K_{p_\theta}$	:	Derivative gain for direction angle controller

where, τ_d is a constant called derivative time.

Between two consecutive target points, (x, y) is the actual position of the vehicle and $(x_{t_{i+1}}, y_{t_{i+1}})$ is the position of the (i+1)th target point where the vehicle is heading for. According to the location of the vehicle relative to the (i+1)th target point, reference direction angle is obtained continuously as follows:

$$\theta_{ref} = \mathrm{atan}\left(\frac{y_{t_{i+1}} - y}{x_{t_{i+1}} - x}\right) \qquad (6.43)$$

Whenever the vehicle is driven for one unit time ΔT, the direction angle reference θ_{ref} is updated based on the present position and direction of the vehicle and the next target point.

Differential steering refers to driving the wheel on one side with more torque than the wheel on the other side. In order to obtain differential steering, PI controller output u_1, and PD controller output u_2 are combined together and the inputs to the right wheel motor and left wheel motor u_R and u_L are formulated , respectively, as follows:

$$\begin{aligned} u_R &= u_1 + \frac{u_2}{2} \\ u_L &= u_1 - \frac{u_2}{2} \end{aligned} \qquad (6.44)$$

As it is seen from equation (6.44), the right wheel motor assumed dominant gives more torque than the left wheel motor.

The vehicle will rotate until the desired direction is obtained while it translates to the target point $(x_{t_{i+1}}, y_{t_{i+1}})$. When the direction angle θ of the vehicle reaches the vicinity of the direction angle reference θ_{ref}, the vehicle will move to the target point $(x_{t_{i+1}}, y_{t_{i+1}})$ only with translational motion. When the vehicle comes to the vicinity of the target point $(x_{t_{i+1}}, y_{t_{i+1}})$, the point is replaced by the next target point. This procedure is repeated till the vehicle reaches the last target point n.

6.5. Simulation of the System

In the simulation of a three-wheeled differentially steered mobile robot, the state space equations which are obtained in 6.3 are solved by using the Runge Kutta-IV numerical integration method. The simulation program is written in the own language of MATLAB 4.0. Changes in state space variables and the motion on x-y plane according to simulation time are shown in plotted diagrams obtained from MATLAB 4.0 graphic display. The flowchart and the text of the simulation program are given in Appendix A and Appendix B, respectively.

The mobile robot's main characteristics employed for simulation are:

$M = 40$ kg ; $I_v = 4$ kg m^2 ; $L = 0.8$m ; $r = 0.06$m

and the following adjusted gain values are used for simulations:

$K_{p_v} = 200.0$; $K_{i_v} = 0.03$

$K_{p_\theta} = 190.0$; $K_{v_\theta} = 28.0$

While adjusting the gains of PI controller, the following effects have been observed:

1. The steady-state error are reduced by increasing the value of the gain K_{p_v}. Increasing this value causes the system response to be more oscillatory. Since the value of the gain K_{p_v} can not be increased too much, it is desirable to modify the proportional controller to a proportional-plus-integral controller.

2. When the integral control action is added to the controller, then as long as there is an error signal, a torque is developed by the controller to reduce the steady-state error.

In the same manner while adjusting the gains of PD controller, the following effects have been observed:

1. Derivative control action, when added to a proportional controller, provides a means of obtaining a controller with high sensitivity. It anticipates the actuating error, initiates an early corrective action, and tends to increase the stability of the system.

2. Derivative control adds damping to the system and thus permits the use of a larger value of the gain K_{p_θ} which will result in an improvement in the steady-state accuracy.

In the first simulation, the mobile robot moves towards only one target point which is (1,1). As it is seen from the Fig. 6.5, the vehicle's initial position is (0,0) and its initial orientation is given equal to the direction angle reference, and the velocity reference is defined as trapezoidal shape. The errors on the state variables are less than 0.001 (with in the units m/sec, rad , and rad/sec for V, $\dot{\theta}$, θ respectively). The vehicle comes to the vicinity of the target point (1,1) with an error which is less than 0.007. This position error is expected because in order to reach the target point, velocity control has been applied.

In the second simulation, as it is seen from Fig. 6.6, the mobile robot moves towards the same target point and the vehicle's initial position is (0,0), but this time, the vehicle's initial orientation is given zero, namely initial pose of the vehicle is (0,0,0). The direction angle reference is generated according to the angle between the actual location of the vehicle and the target point where the vehicle heads for. The velocity reference is defined as trapezoidal shape. The errors on the state variables are less than 0.001. The vehicle comes to the vicinity of the target point (1,1) with an error of 0.0082 which is a little bit bigger than the previous example. This is expected because while the vehicle translates to the target point (1,1), it tries to turn its direction θ to the direction angle reference θ_{ref}.

In the third simulation, as it is seen from Fig. 6.7, the mobile robot moves towards five target points which generates a $y=x^2$ parabolic function. The vehicle's initial pose is (0,0,0). The direction angle reference is generated continuously according to the angle between the actual location of the vehicle and the target point where the vehicle heads for in each unit time ΔT. The velocity reference is defined as trapezoidal shape and generated in each unit time ΔT according to the distance where vehicle heads for. The errors on the state variables are less than 0.001. The vehicle does not stop at the transient target points, it stops when it comes to the vicinity of the last target point (5,25) with an error less than 0.01 which is acceptable.

In the fourth and the last simulation, the aim is to show an important feature of differential steering. To allow the robot to turn in even tighter quarters, one front wheel is driven forward and the other is in reverse. The vehicle would then turn around the center point between these two front wheels. As it is seen from Fig. 6.8, the vehicle only rotates in order to reach direction angle reference and does not translate. The actual velocity of the vehicle is zero and the vehicle stays in its own position (0,0). The steady state error is less than 0.001.

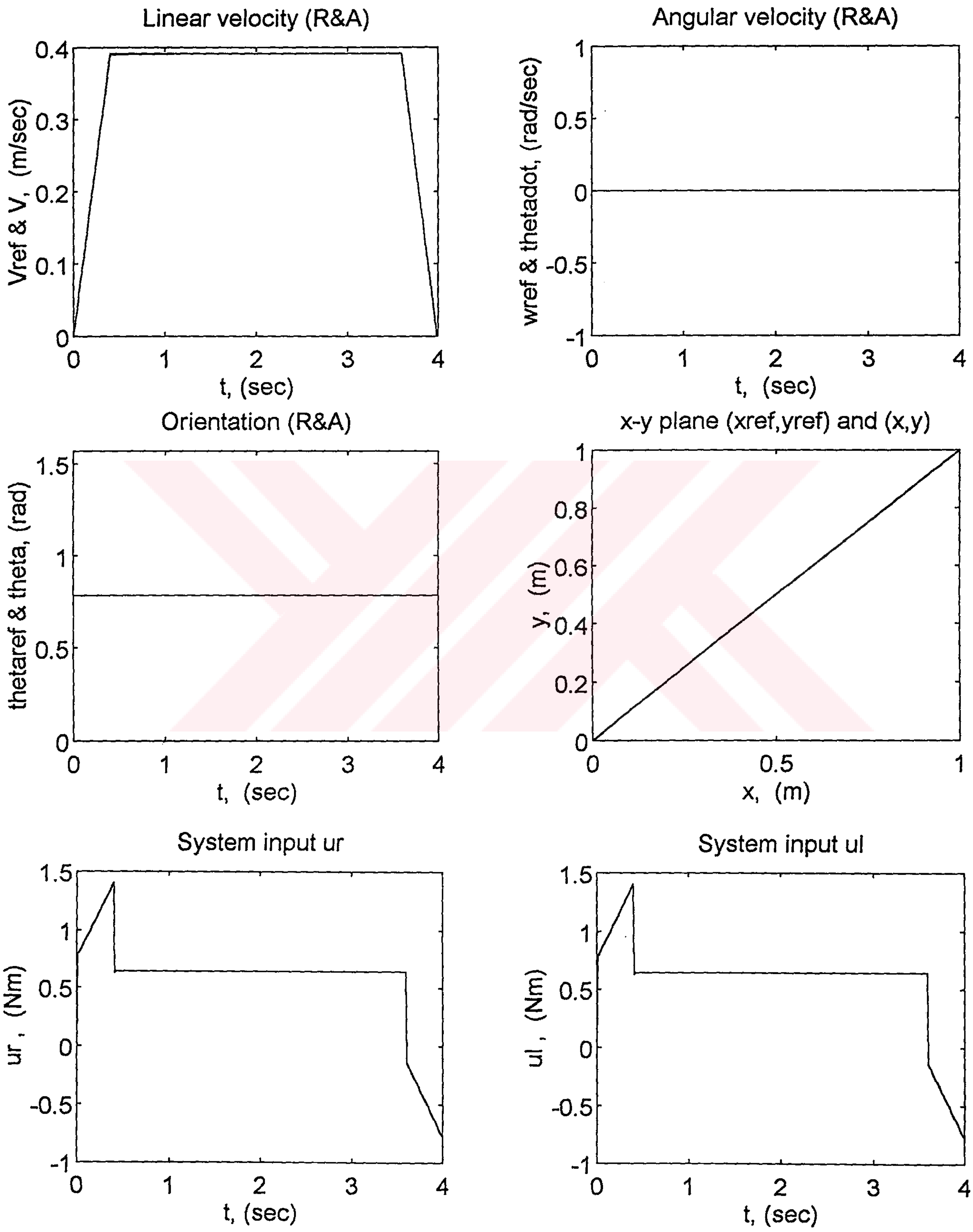


Figure 6.5 Moving to only one target point, ($\theta_o=\theta_{ref}$)

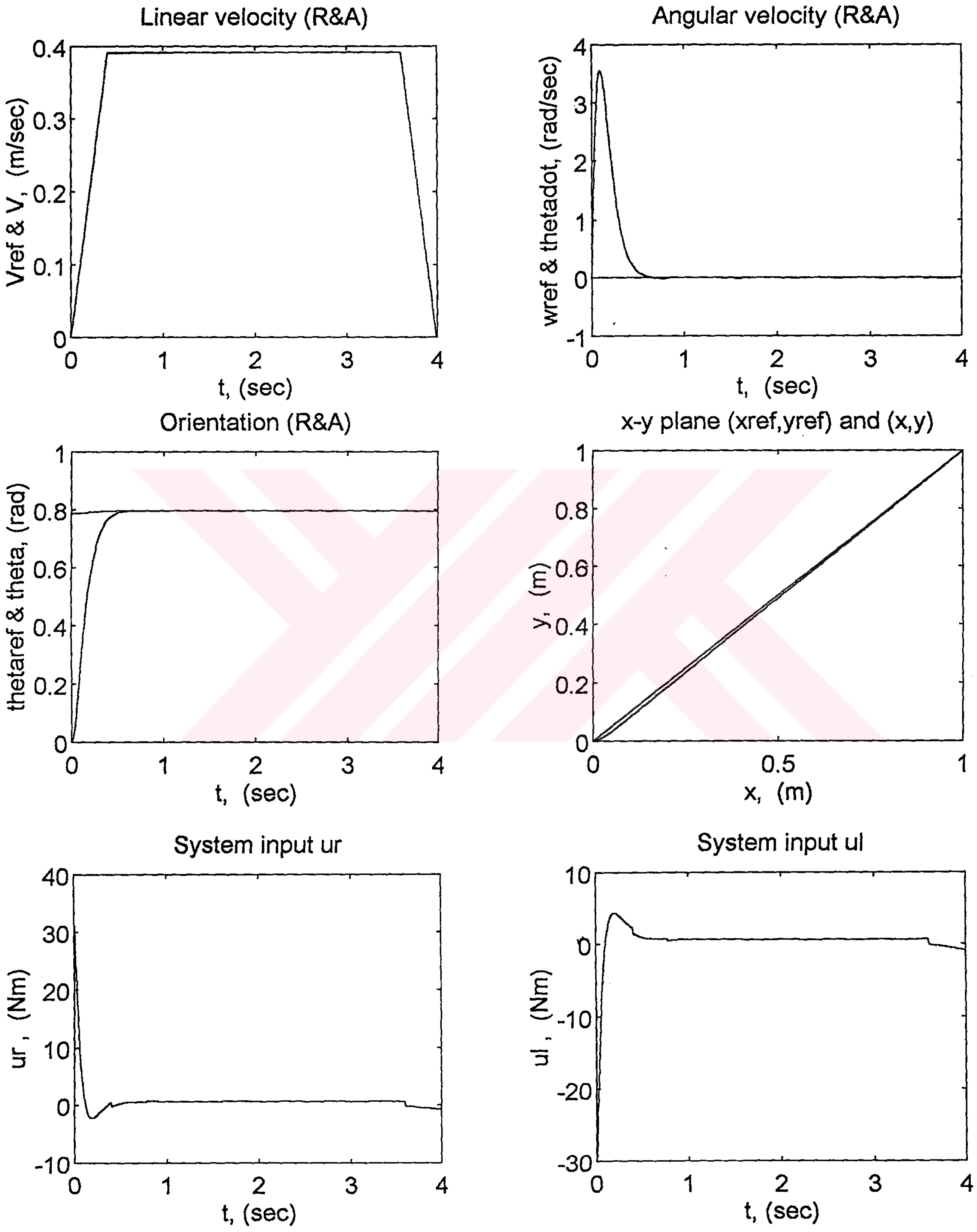


Figure 6.6 Moving to only one target point, ($\theta_o = 0$)

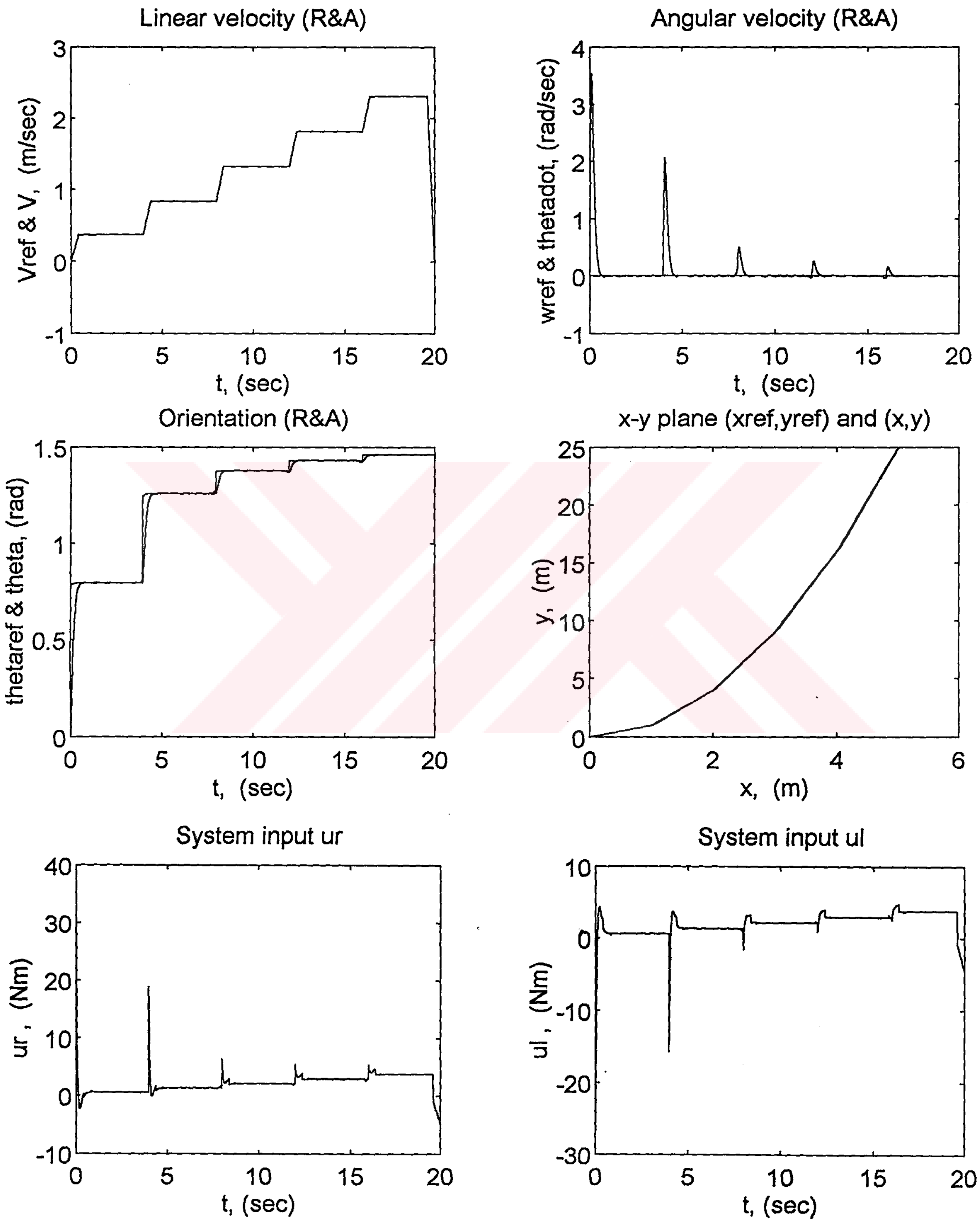


Figure 6.7 Moving to several target points which generates a parabolic trajectory

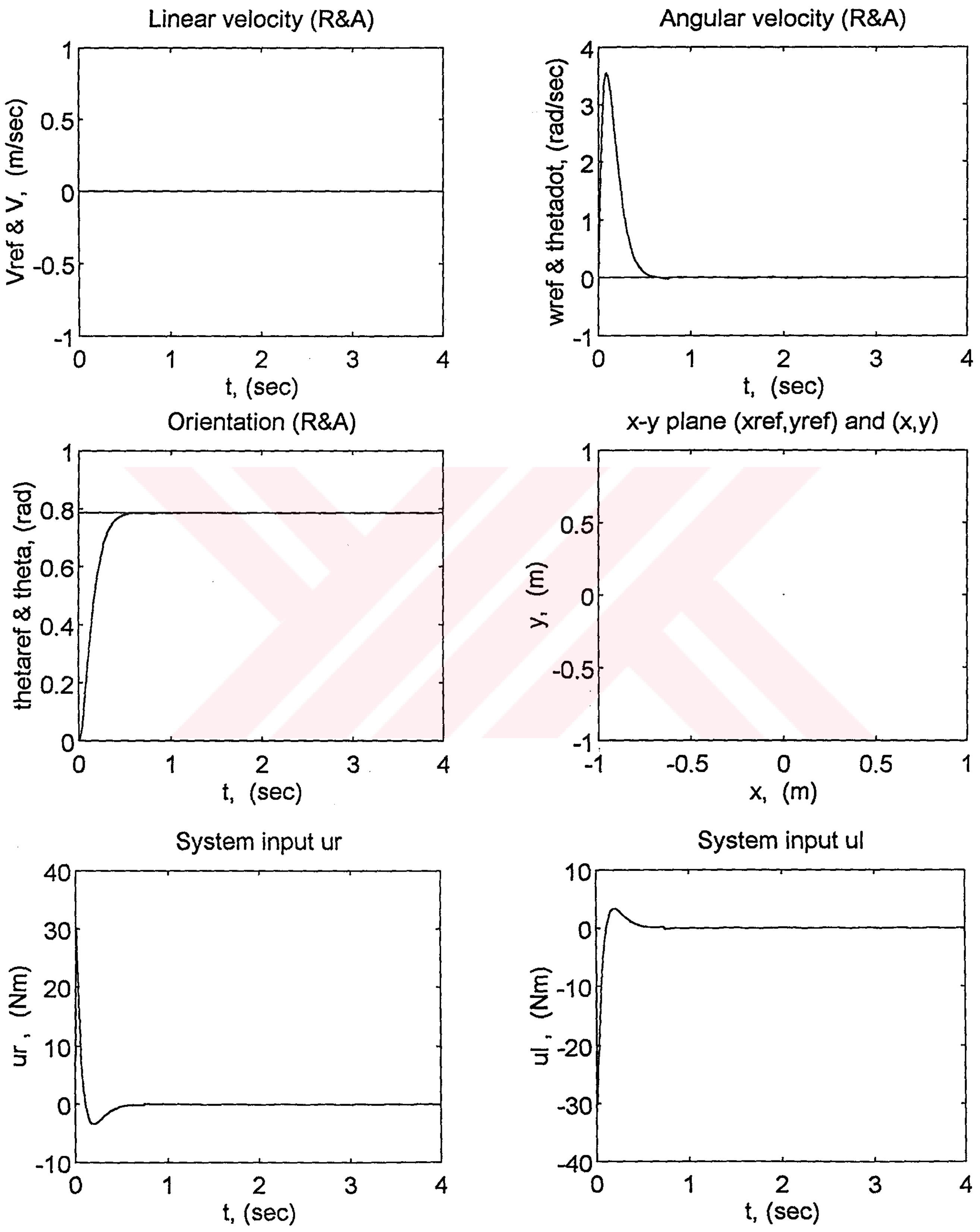


Figure 6.8 Turning on a dime, ($\theta_o = 0$)

RESULTS

As a result, there are a large variety of mobile robot applications in which a robot will operate in large and unstructured domains, for example, delivering parts in a large factory from an automated warehouse to an automated workcell, fire fighting, demolishing structures, cleaning industrial waste sites, maintaining nuclear plants, inspecting and repairing underwater structures, assembling structures in outer space, cleaning windows and aiding the handicapped. Mobile robots operating in such large, unstructured domains must be able to cope with significant uncertainty in the position and identify the objects. In fact, the uncertainty is such that one of the most challenging activities for a mobile robot is simply going from point S to point G, a relatively trivial activity for an industrial manipulator. As if in compensation for having to cope with more uncertain environments, it is not expected for a mobile robot to follow paths or to reach destinations at the same level of precision that is expected from an industrial manipulator.

Most desirable type of mobile robot is completely autonomous. As a class of robotic system, autonomous mobile robots constitute one of the important steps in the evolution of robotic intelligence and structure. The autonomous mobile robot should be able to formulate and generate explicit action plans using an implicit goal or mission description as input. It should be able to execute and supervise explicit action plans independently and understand its world and to derive coherences with the current action. Communication and interaction with its world is an important factor. It should be capable to react to unforeseen events or situations and to learn for skill acquisition or internal model building.

In the design of mobile robots, the form of locomotion utilized by the robot must first carefully selected. As this selection of the type of locomotion is crucial to the performance of the robot, poor selection can make success in the design of the mobile robot near to impossible. In theory, the choice depends upon trade-offs in power requirements, costs, desired locomotion characteristics, and the surfaces which the robot must transverse. In practice, however, the choice is usually wheels. Because there are several engineering problems remain unsolved in legged mobile robots. Wheels perform well if the terrain is level or has only a limited slope, is reasonably smooth and good traction . Otherwise tracks can be used, they are well suited for outdoor, off-road applications. Legged mobile robots are difficult to design, the computations required for leg control and balance are numerous and difficult and their power drain is so large that these robots are currently seen as only experimental models in the university or R&D Laboratories. But the legged locomotion has special characteristics such as the greatest capability of going over most types of terrain and climbing stairs. A smooth ride can be achieved on rough ground; a leg does not like a wheel, waste power by always having to climb out of a rut of its own making and is less susceptible to digging deeper and deeper until the vehicle stops and feet may do less damage to the ground than tracks or wheels. The articulated body mobile robot can pass over uneven terrain and narrow paths by actively adapting its long trunk to the ground topography.

The actual design of the navigation systems obviously depends upon accuracy requirements, mission environments, usage time, reaction time, reliability, cost and many other factors. Odometry is simple and inexpensive, however, this method always shows a steadily increasing error due to wheel slippage and tyre elasticity. Surface roughness and undulations may cause the distance to be overestimated, wheel slippage can cause the distance to be underestimated. Therefore the mobile robot needs to call at intervals at known locations at which the navigation system can be reset. The inertial navigation sensors although currently expensive are expected to decline in price, while performance is expected to improve steadily, especially for ring laser gyros. In the future applications, it can be expected conventional inertial guidance sensors to be increasingly used. Vision and direct imaging systems provide information that is difficult to obtain in other ways. Although a few vision systems have been available for about ten years, they have not been used because of their limited capabilities and high cost. However vision systems as mobile robot navigation systems are now used more and more frequently because of the development in real-time vision systems and decreasing prices. The main disadvantage of vision and direct imaging systems is that they require complex and time consuming analysis and computations which restrict the velocity of the mobile robot. Compared to vision systems, ultrasonic systems are less expensive, faster, but they have some shortcomings. Their poor directionality limits the accuracy in determining spatial position of an edge to 10-50cm, depending on the distance to the obstacle and the angle between the obstacle surface and the acoustic axis. Frequent misreadings are caused by either ultrasonic noise from external sources or stray reflections from neighboring sensor. Specular reflections occur when the angle between the wavefront and the normal to a smooth surface is too large. In this case, the surface reflects the incoming ultrasound waves away from the sensor. The beacons which are used in triangulation can be infrared, ultrasonic, laser, or even bar-codes placed at strategic positions. Although very accurate, the beacons which are used in triangulation method have a few problems. Several must be placed wherever the robot may wish to go, thus increasing total system cost. Triangulation method also requires real-time performance. An another disadvantage is that it requires unobstructed line-of-sight which is not always possible. Environment dependent sensory driven navigation (multisensory navigation) is at present the key issue in mobile robots research and this is still more the case when real world applications are concerned. Generally, most researchers have used dead reckoning method coupled with periodic calibration using one of the given methods for navigation of mobile robots. Although this approach seems to be quite effective, more needs to be done to develop a reliable and accurate navigation system for real world implementations. The state-of-the-art of navigation system technology today is still at an infancy stage. In order to have widespread application in the future, sensory capabilities required would include :

1. Development of adequate mechanisms for sensor interpretation.
2. Generation of world models from sensory information, leading to robust and useful representations for robotic spatial-reasoning tasks.
3. Integration of multiple sensory-information sources, provided either by similar or by qualitatively different sensors, into a uniform and coherent description of the robot's environment.

In path planning, it is assumed that the surfaces of the moving object and of the obstacles are algebraic. Because full information is assumed, the whole operation of path planning is a one-time, off-line operation. The main difficulty is not in proving that an algorithm that would guarantee a solution exists, but in obtaining a computationally efficient scheme. Conceptually, cases of arbitrary complexity can be considered, given the fact that a solution is always feasible. The advantage of dealing with complete information is that any optimization criteria such as the shortest path or the minimum-time path or the safest path can be easily introduced. In path planning with complete information approach, some approximations are made in the algorithms. A slight change in the specified accuracy of the approximation can cause a dramatic change in the approximated surfaces and eventually in the generated paths. During the past years, Potential Field Methods for obstacle avoidance have gained increased attention from researchers in the field of robotics and mobile robots. One of the reason for the popularity of this method is its simplicity and elegance. It can be implemented quickly and initially provide acceptable results without require refinements. Also, no expensive precomputation as for C-space or Voronoi diagram is necessary. A smooth path is automatically created. As this approach uses only local information which could be obtained by sensors, it is fast enough to run in real-time. However the problem with this strictly local method is that the planner might get stuck in a local minimum and would not reach the final configuration.

The control structure of an autonomous mobile robot should be programmable, autonomous, robust, reactive and adaptive. There are essentially two forms of control architecture in use or in development at the present time. Firstly, there is the Functional Decomposition in which the mobile robot is considered as a group of processes, known as agents, each carrying out a specific function. The structure of Cognitive Controller is being decomposed into the three levels of control: Planner, Navigator and Pilot. It has a strictly hierarchical architecture and the higher the level of hierarchy the more is the time lag between the motion control determined and actually executed. The Distributed Control Architecture embodies a hierarchical model of distributed computation. This shows the decision-making model. Control decisions are, whenever possible, made locally in the module which is confronted with a problem. Otherwise, the problem or data is broadcast recursively to the next higher level of decision making. Another result from this control structure is that commands and data can exist within the system at several levels of abstraction. At each level , only the necessary degree of detail is present. Higher levels are able to deal with the same information in a more abstract form and do not become cluttered with unnecessary details. The system is loosely coupled since the rate of communication between modules, especially beyond the motor and sensor subsystem levels, is relatively small. This results from the use of asynchronous processes and the Blackboard. Such an approach lead to higher performance and better adaptability to dynamically changing conditions. Hilare Control Architecture has a fixed hierarchical structure capable to general planning and execution monitoring. There are resources for routing, navigation, sensors, which are used to produce a linear plan and control its execution. The controller is organized by a rule-based system and control structure uses the concept of process to achieve a controlled reactivity through the use of surveillance monitors providing a flexible control of these processes. It has reactivity and error recovery capacities.

The second control structure is the Behavioral Decomposition which considers the mobile robot to be composed of levels of behavior such as avoid the obstacles, wander around, explore the environment, etc. The Layered Control Structure which is an application of Behavioral Decomposition decomposes the control system into a set of loosely coupled task achieving behaviors. Each behavior is a complete control system going from sensory inputs to motor outputs. Once a behavior is debugged, it is no more changed. Sophisticated control systems can be build around it because its simple, extensible arbitration scheme. Both functional and behavioral control decompositions have their advantages and disadvantages. The behavioral decomposition and state transition layer are essential in the design of the control strategy, whereas the functional or hierarchical decomposition is important in the physical realization.

Although there has been significant amount of work on the control of robot manipulators, there has been little theory on the control of autonomous mobile robots, presumably because of the newness of the field. The PTP method is used to control the vehicle to follow transient target points. For the PTP motion control of the vehicle, there are two degrees of freedom to control, that is, direction angle and the velocity of the vehicle. For direction angle (orientation) control of the vehicle, PD controller and for linear velocity control of the vehicle, PI controller are used. Between two consecutive target points, (i) and (i+1), the vehicle rotates until the desired direction is obtained while it translates to the (i+1) target point. When the direction angle θ of the vehicle reaches the vicinity of the direction angle reference θ_{ref}, the vehicle moves to the target point (i+1) only with translational motion. When the vehicle comes to the vicinity of the target point (i+1), the point is replaced by the next coming target point. This procedure is repeated till the vehicle reaches the last target point n. According to the results of simulations, the control algorithm enables the vehicle to reach target points given in a different trajectories with an acceptable position error. This error is expected because in the control algorithm instead of position control, velocity control has been applied. Between two consecutive target points, the vehicle decides its speed according to the distance which will move. Two kinds of motion are applied simultaneously, namely rotational and translational motions. The control strategy could be also designed to perform only two distinct kinds of motion: straight-line motion, where both motors are running at the same speed and in the same direction, and rotation about the vehicle's center point, where both motors are running at the same speed but in opposite directions. This kind of control strategy is relatively simple, but distinct kinds of motion is not desirable when time and continuos motion are considered. At the last simulation in order to show the feature of differentially steered vehicle, the vehicle does not translate only rotates around its center point. The vehicle can turn in even tighter quarters for that reason, differentially steered three-wheeled mobile robots are very popular in in-door applications. In this study PTP method has been applied, in the next projects continuous path (CP) algorithms can be searched. The CP method would be more smoother and it would be easier to change the traveling course during real time applications. In PTP method, it is necessary to generate the series of transient target points.

REFERENCES

[1]KHOUKHI A., HAMAM Y., “Mobile Robot Kınematic, Dynamic, and Mobility”, IEEE Int. Conference on Advanced Robotics, Vol. 2, Pisa,1991

[2]POOLE H. H.,”Fundamentals of Robotics Engineering” Van Nostrand Reinhold Ltd., 1989.

[3]KOMORİYA K., TANI K.,” Utilization of the Virtual Environment System for Autonomous Control of Mobile Robots”, IEEE Int. Workshop on Intelligent Motion Control, Istanbul, August 1990.

[4]WILLIAMS J. D., “ AGVs 2000: What Lies Ahead”, Automation, pages 38-40, June 1991

[5] MEYSTEL A., “ Autonomous Mobile Robots”, World Scientific Publication Co. Pte. Ltd., 1991

[6]DILLMAN R, “ Mobile robots in industrial environment”, Proc. Int. Symposium on Industrial Robots, pp, 79-90, April 1988

[7]IYENGAR S., ELFES A., “Autonomous Mobile Robots: Control, Planning, and Architecture, IEEE Computer Society Press,1991

[8]WILCOX B.H., GENNERY D.B.,” A Mars Rover for the 1990’s ”, Journal of the British Interplanetary Society, Vol.40, 1987

[9]PREMVUTI S., YUTA S.,” Consideration on the Cooperation of Multiple Autonomous Mobile Robots”, IEEE Int. Workshop on Intelligent Robots and Systems, IROS ’1990.

[10] WEISBIN, G., EINSTEIN J.R.,” Autonomous Mobile Robot Navigation and Learning”, IEEE Computer Magazine, pages 29-35, 1989

[11]REMBOLD U., “ The Karlsruhe Autonomous Mobile Asembly Rbot”, Proceedings of the IEEE International Conference on Robotics and Automation,pg 598-603,1988

[12]NELSON W.L., COX I.J.,” Local Path Control for an Autonomous Vehicle”, Proceedings of the 1988 IEEE International Conference on Robotics and Automation, Philadelphia, April 1988.

[13]KRIEGMAN D. J., TRIENDL E., BINFORD T.O., " A Mobile Robot: Sensing, Planning and Locomotion", IEEE Int. Conference on Robotics and Automation, 1987

[14] TACHI S., KOMORIYA K., " Guide Dog Robot", International Symposium on Robotics Research, Vol.2 .

[15] EVANS J., KRISHNAMURTHY B., BARROWS B., SKEWIS T., LUMELSKY V., " Handling Real-World Motion Planning: A Hospital Transport Robot", IEEE Int. Conference on Robotics and Automation, Sacramento, April 1991.

[16] SIMONS G.," Robots: The Quest for Living Machines", Cassel Book, 1992

[17] TODD, D.J., " Fundamentals of Robot Technology", Kogan Page Ltd, 1986

[18] ROSHEIM M.E., "Robot Evolution: The Development of Anthrobotics", John Wiley & Sons Inc.,1994

[19] WILCOX H., GENNERY D.," Mobile Robots II" ,SPIE, 1987

[20] NONOSE S., TAISUKE O.," Total System of Advanced Subsea Robot", IEEE Int. Conference on Advanced Robotics, Vol. 2 , Pisa,1991.

[21]HIROSE S., " Three Basic Types Of Locomotion in Mobile Robots", IEEE Int. Conference on Advanced Robotics, Vol. 1, Pisa,1991

[22] MUIR P.F., NEUMAN C.P., " Kinematic Modelling of Wheeled Mobile Robots", The Journal of Robotic Systems, Vol.4, pages 281-340,1987

[23] RAIBERT M.H., "Legged Robots", Robotics Science, 1989

[24]RAIBERT M. H., " Running With Symmetry", The International Journal of Robotics Research, Vol. 5, 1986

[25]RAIBERT M.,"Troting, Pacing and Bounding by a Quadruped Robot.", Journal of Biomechanics, pages. 79-98, 1990

[26]HIROSE S.,"A Study of Design and Control of a Quadruped Walking Vehicle ", International Journal of Robotics Research, Vol. 3,1984

[27]MCGHEE R., WALDRON K.," The Adaptive Suspension Vehicle Project: A Case Study in the Development of an Advanced Concept for Land Locomotion.", Unmanned Systems,1985

[28]RUSSELL M.," Odex 1: The First Functionoi.", Robotics Age, pages 12-18, 1983

[29]BARES J., HEBERT M., KAADE T., KROTKOV E., "Ambler: An Autonomous Rover for Planetary exploration", IEEE Computer Magazine, pages 18-26, June 1989

[30]HIROSE S., MORISHIMA A., TUKAGOSI S., " Design of Practical Snake Vehicle: Articulated Body Mobile Robot KR-II", IEEE Int. Conference on Advanced Robotics, Vol. 1, Pisa,1991

[31] YOSHIAKI I., NORIHIKO O., KENICHERO S.," A Hybrid Locomotion Vehicle for Nuclear Power Plants", IEEE Trans. System, Man and Cybernatics,Vol.SMC-13, 1983

[32] COX I. J., " BLANCHE: An Experiment in Guidance and Navigation of an Autonomous Robot Vehicle.", IEEE Transactions on Robotics and Automation, Vol. 7, 1991

[33] HONGO T., ARAKAWA H., SUGIMOTO G., TANGE K., " An Automatic Guidance System of a Self-Controlled Vehicle", IEEE Transactions on Industrial Electronics, 1987

[34] KURITSKY M. M., Goldstein M. S. " Inertial Navigation" Proceedings of the IEEE, Vol.71,1983

[35] BATEMAN P.J., " Guidance and Control for Unmanned Ground Vehicles", AGARD Lecture Series, June 1994

[36] MORAVEC H.P.," The Stanford Cart and CMU Rover", Proceedings of the IEEE, Vol.71,1983

[37] MILLER G. L., WAGNER E.R., " An Optical Rangefinder for Autonomous Robot Cart Navigation",Proceedings of the SPIE, Vol.852, Mobile Robots II, 1987

[38] GONZALEZ J., OLLERO A., HURTADO P., " Local Map Building for Mobile Robot Autonomous Navigation by Using a 2D Laser Range Sensor", IEEE Transactions of Robotics and Automation, Vol.9, 1992.

[39] THORPE C., HEBERT M.H., KANADE T. ,SHAFER S.A.," Vision and Navigation for the Carneige-Mellon Navlab", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.10, May 1988.

[40] TURK M.A., MORGENTHALER D.G., GREMBAN K.D., MARRA M., " A Vision System for Autonomous Land Vehicle Navigation", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.10, May 1988

[41] DICKMANNS E.D., GRAEFE V., "Machine Vision and Applications", Springer-Verlag New York Inc.,1988

[42] GRAEFE V., " Two multi-processor systems for low-level real-time vision", Robotics and Artificial Intelligence, Springer-Verlag, Berlin, 1984

[43] EVERETT H. R., " A Multi-Element Ultrasonic ranging Array", Robotics Age, Vol.7, July 1985

[44] ELFES A.," Sonar-Based Real-World Mapping and Navigation", IEEE Journal of Robotics and Automation Vol.RA-3,1987

[45] TORRIERI D. J.," Statistical Theory of Passive Location Systems", IEEE Transactions on Aerospace and Electronic Systems, Vol. AES-20,1984

[46] KOSKINEN K., MAKELA H.," Navigation of Outdoor Mobile robots Using Dead Reckoning and Visually Detected Landmarks", IEEE Int. Conference on Advanced Robotics, Vol. 2, Pisa,1991

[47] MAYBECK P.S. " Stochastic Models, Estimation and Control", Academic Press,1979

[48] GIRALT G., CHATILA R., VAISSET M., " An Integrated Navigation and Motion Control System for Autonomous Multisensory Mobile Robots", International Symposium on Robotics Research, pages 191-214, 1984

[49] EVENS J., " HELPMATE: A Service Robot for Healthcare",Industrial Robot, June 1989 .

[50] LOZANO-PEREZ T., " Spatial Planning: A Configuration Space Approach", IEEE Transactions on Computers, Vol. C-32, 1983

[51] LUMELSKY V.J., STEPANOV A.A.," Algorithmica", Springer-Verlag New York Inc., 1987

[52] SCHILLING R.J., " Fundamentals of Robotics: Analysis and Control ", Prentice-Hall Inc.,1990

[53] BROOKS R. A., " Solving the Find-Path Problem by good Representation of free Space", IEEE Transactions on Systems, Man, and Cybernetics, Vol.SMC-13,1983

[54] CANNY J., DONALD B.," On the Complexity of Kinodynamic Planning", In Proceedings of the 29th IEEE Symposium on foundations of Computer Science, New York, 1988.

[55] BARHOLOMEW N.,"Theory of Automatic Robot Assembly and Programming", Chapman&Hall,1993

[56] DYM L.C., LEVITT E.R., "Knowledge-Based Systems in Engineering", McGraw-Hill International Editions,1991

[57] KHATIB O., " Real-Time Obstacle Avoidance for Manipulators and Mobile Robots", The Int. J. of Robotics Research, Vol.5,1986

[58] CLAVEL N., THOMPSON P., SEVILA F., ZAPATA R., " A Mobile Patform with a Robotic Arm Working in a Semi-Structured Environment: Two basic Contributions", IEEE Transactions of Robotics and Automation, Vol.5,1992.

[59] KOREN Y., BORENSTEIN J., " Analysis of Control Methods for Mobile Robot Obstacle Avoidance", IEEE International Workshop on Intelligent Motion Control, Istanbul, August 1990.

[60] KOREN Y., BORENSTEIN J.," The Vector Field Histogram - Fast Obstacle Avoidance for Mobile Robots", IEEE Transactions on Robotics and Automation, Vol.7, August1991.

[61] CAMARGO R. F., CHATILA R., ALAMI R., " A Distributed Evolvable Control Architecture for Mobile Robots", IEEE Int. Conference on Advanced Robotics, Vol. 2, Pisa,1991

[62] ELFES A., " A Distributed Control Architecture for an Autonomous Mobile Robot", Int. J. Artificial Intelligence in Eng., Vol.1,1986

[63] NOREILS R. F., CHATILA R., " Plan Execution Monitoring and Control Architecture for Mobile Robots", IEEE Transactions on Robotics and Automation, Vol.11, April 1995.

[64] NOREILS R. F., CHATILA R., "Control of Mobile Robot Actions", Proceedings of the IEEE International Conference on Robotics and Automation, 1989

[65] BROOKS R. A.," A Robust Layered Control System for a Mobile Robot", IEEE Journal of Robotics and Automation, Vol.RA-2, March 1986.

REFERENCES NOT CITED

[1] KOIVO A.J., " Fundamentals for Control of robotics Manipulators", John Wiley and Sons, 1989.

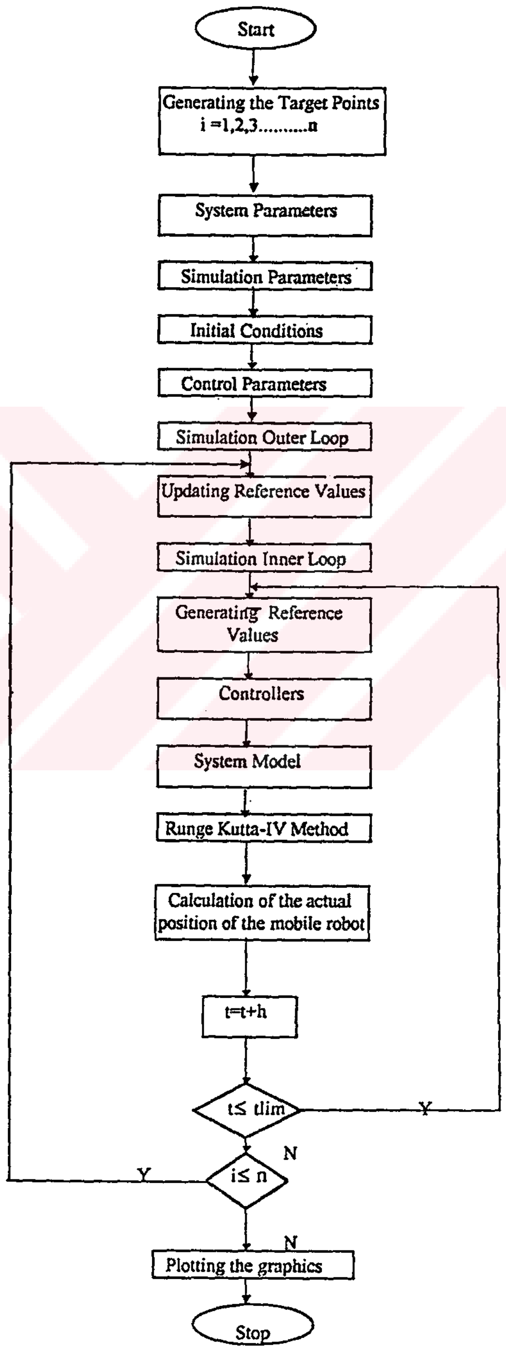
[2] The MATH WORKS, Inc., " The Student Edition of Matlab", Prentice-Hall, Inc., 1992.

[3] MELIKŞAH E.," Control of Automated Guided Vehicles",M.Sc Thesis, BU, 1994.

[4] TSUGAWA S., HIROSE T., YATABE T., " An Intelligent Vehicle with Obstacle Detection and Navigation Functions", Int. Conference on Industrial Electronics, Control and Instrumentation, 1984.

[5] KOREN Y., BORENSTEIN J., " Motion Control Analysis of a Mobile Robot", Journal of Dynamic Systems, Measurement, and Control, Vol.109, June 1987.

APPENDIX A SIMULATION FLOWCHART



APPENDIX B SIMULATION PROGRAM

```
%	Generating the target points subroutine			(nodes.m)

xkonum(1)=0.0; ykonum(1)=0.0;
xkonum(2)=1.0; ykonum(2)=1.0;
xkonum(3)=2.0; ykonum(3)=4.0;
xkonum(4)=3.0; ykonum(4)=9.0;
xkonum(5)=4.0; ykonum(5)=16.0;
xkonum(6)=5.0; ykonum(6)=25.0;

nodenum=5;


%	System parameters subroutine		(sysparam.m)

M=40;  r=0.06;  Iv=4;	Iw=0.8;		Im=0.0002;	n=20;  L=0.8;

k1=4.042e-1;  k2=4.8e-3;	Bw=0.01;	bm=0.0001;

A=(M*r+2*Iw/r+2*Im*n^2/r);
B=(2*k2*n^2/r+2*Bw/r+2*bm*n^2/r);
C=k1*n;
D=(2*Iv*r/L+Iw*L/r+Im*n^2*L/r);
E=(k2*L*n^2/r+Bw*L/r+bm*n^2/r);
F=k1*n;


%	Simulation parameters subroutine		(simparam.m)

h=0.01;		dt=h;  t0=0.;  tlim=0.99;	nstep=((tlim-t0)/h)+1;

tolorient=0.001;	tolspeed=0.001;

k=1;  j=1;


%	Initial conditions subroutine		(initcond.m)

x(1)=0.0;	x(2)=0.0;	x(3)=0.0*pi/180;

%velocity	%thetadot	%theta
```

```
x1(1)=x(1);    x2(1)=x(2);    x3(1)=x(3);

vx(1)=0.0;     vy(1)=0.0;
wr(1)=0.0;     wl(1)=0.0;
xk(1)=0.0;     yk(1)=0.0;

status=1;

xk(1)=xkonum(j);      yk(1)=ykonum(j);


U1=0;  U2=0;

SigmaSpeedErr=0;

vref=0;

z=vref;


%       Control parameters subroutine                      (cntparam.m)

Kpvelo=200.0;                    Kivelo=0.03 ;

Kporient=190.0;          Kvorient=28.0;


%       Calculate reference max velocity subroutine   (calcmvref.m)

vref1=direction*(deslength)/(tlim-t0);


%       Generate reference velocity subroutine                       (refvelo.m)

if  k<=((j-1)*nstep+(nstep/10))


        vref=vref+(vref1-z)/(nstep/10);


end

if k>((j-1)*nstep+(nstep/10)) & k<=(j*nstep-(nstep/10))

        vref=vref1;

end
```

```
if j==nodenum

          if k>(j*nstep-(nstep/10))

                    vref=vref-(vref1-0)/(nstep/10);

                              if abs(vref)> abs(vref1)
                                        vref=0.0;
                              end
          end

else

          if k>(j*nstep-(nstep/10))

                    vref=vref1;

          end

end

a(k)=vref;


%         Generate reference orientation subroutine                (desorien.m)

if abs(xk)<abs(xref) & abs(yk)<abs(yref)

          desorient(k)=atan2((yref-yk(k-1)),(xref-xk(k-1)));


else

          desorient(k)=desorient(k-1);

end


bil(k)=desorient(k);

if desorient(k)<0 & desorient(k)>-pi

          bil(k)=desorient(k)+pi;

end
```

```
%       Calculate orientation error subroutine          (calcdifo.m)

        fullpi=2*pi;
        a1=x(3) / fullpi;
        b1=(a1-fix(a1))*fullpi;
        a2=desorient(k) / fullpi;
        b2=(a2-fix(a2))*fullpi;
        c(k)=b1;
        diforient=b2-b1;
        erteta(k)=diforient;


%       Find direction and pole subroutine              (direpole.m)

        if cos(diforient)>0
                direction=1;
        else
                direction=-1;
        end
        if tan(diforient)>0
                pole=1;
        else
                pole=-1;
        end


%       PI Velocity controller subroutine               (velocont.m)

ev(k)=vref-x(1);
DeltaU1=Kpvelo * (ev(k)-ev(k-1)) + Kivelo * (ev(k)+ev(k-1))/(2*h);
U1=U1 + DeltaU1;


%       PD Orientation controller subroutine            (oriecont.m)

        if abs(diforient)>(pi/2)
                diforient=abs(abs(diforient)-pi);
        end

        U2=Kporient*abs(diforient) - pole*Kvorient*x(2);


%       System model subroutine                 (system.m)

dx(1)=-(B/A)*x(1)+(C/A)*((ur+ul));
dx(2)=-(E/D)*x(2)+(F/D)*((ur-ul));
dx(3)=x(2);
```

```
%       Runge Kutta-IV method subroutine            (runge4.m)

for i=1:3
        b(i)=x(i);
end

system;

for i=1:3
        p1(i)=h*dx(i);
        x(i)=b(i)+0.5*p1(i);
end

system;

for i=1:3
        p2(i)=h*dx(i);
        x(i)=b(i)+0.5*p2(i);
end

system;

for i=1:3
        p3(i)=h*dx(i);
        x(i)=b(i)+p3(i);
end

system;

for i=1:3
        p4(i)=h*dx(i);
        x(i)=b(i)+(p1(i)+2.0*p2(i)+2.0*p3(i)+p4(i))/6.0;
end

end

%       Calculate the actual position of the mobile robot subroutine  (calcul.m)

x1(k)=x(1);    x2(k)=x(2);    x3(k)=x(3);

vx(k)=x1(k)*cos(x3(k));
vy(k)=x1(k)*sin(x3(k));

xk(k)=xk(k-1)+(dt/2.0)*(vx(k)+vx(k-1));
yk(k)=yk(k-1)+(dt/2.0)*(vy(k)+vy(k-1));
```

```
%       Plotting the graphics subroutine                  (graphic.m)


figure(1)


subplot(221)
plot(t,a,t,x1)
title('Linear velocity (R&A)');
xlabel('t, (sec)')
ylabel('Vref & V,  (m/sec)')


subplot(222)
plot(t,wref,t,x2)
title('Angular velocity (R&A)');
xlabel('t,  (sec)')
ylabel('wref & thetadot, (rad/sec)')


subplot(223)
plot(t,bil,t,x3)
title('Orientation (R&A)');
xlabel('t,  (sec)')
ylabel('thetaref & theta, (rad)')

subplot(224)
plot(xkonum,ykonum,xk,yk)
title('x-y plane (xref,yref) and (x,y)');
xlabel('x,  (m)')
ylabel('y,  (m)')

figure(2)

subplot(221)
plot(t,signalr)
title('System input ur');
xlabel('t, (sec)')
ylabel('ur ,  (Nm)')


subplot(222)
plot(t,signall)
title('System input ul');
xlabel('t, (sec)')
ylabel('ul ,  (Nm)')
```

```
%       Main Program

clear

nodes;          % Generate the target points

sysparam;       % System parameters

simparam;       % Simulation parameters

initcond;       % Initial conditions

cntparam;       % Control parameters

% Start of simulation loop   (Outer loop)

for j=1:nodenum

%       Updated reference values

xref=xkonum(j+1);    yref=ykonum(j+1);

deslength=sqrt((xref-xk(k))^2+(yref-yk(k))^2);

desorient(k)=atan2((yref-yk(k)),(xref-xk(k)));

bil(k)=desorient(k);

if desorient(k)<0 & desorient(k)>-pi

        bil(k)=desorient(k)+pi;

end

calcdifo;       % Calculate orientation error

direpole;       % Find direction and pole

calmvref;       % Reference max. velocity calculation
```

```
wref(k)=x2(1);

z=vref;

%       Start of inner loop

ksim=k;

for ksim=k+1:nstep*j

        k=ksim;

        refvelo;                % Generate reference velocity

        desorien;               % Generate reference orientation

        wref(k)=x2(1);

        calcdifo;               % Calculate orientation error

        direpole;               % Find direction and pole

% Velocity & Orientation Controllers

        if abs(x1(k-1)-vref)>tolspeed

                        velocont;

        end

        if abs(diforient)>tolorient

                oriecont;

        end

        ur=U1+pole*(U2/2);
        ul=U1-pole*(U2/2);

        signalr(k)=k1*ur;
        signall(k)=k1*ul;
        runge4;                         % Runge Kutta-IV method

        calcul;                         % Calc. the position of the vehicle

end

% Updated values

x1(k+1)=x1(k);
x2(k+1)=x2(k);
x3(k+1)=x3(k);
xk(k+1)=xk(k);
yk(k+1)=yk(k);
desorient(k+1)=desorient(k);
a(k+1)=a(k);
bil(k+1)=bil(k);
wref(k+1)=wref(k);
signalr(k+1)=signalr(k);
signall(k+1)=signall(k);

end


% Graphical display

xk=xk';
yk=yk';
t=0:dt:k*h;

graphic

end
```

CURRICULUM VITAE

Bilin Aksun was born in Istanbul, on July 31, 1972. She graduated from Şişli Terakki Highschool with high honour degree (first runner up) and received an award for foreign language (English) in 1989. She received the B.Sc. in Mechanical Engineering from Istanbul Technical University in 1993 and began M.Sc. at Robotics Program of Istanbul Technical University Science and Technology Institute in the same year. Since January 1994, she has been working as a research assistant in Automatic Control Laboratory at Istanbul Technical University Department of Mechanical Engineering.