Beş Eksenli Bir Robot Kolu Simülasyonu ve Engel/Hedef Takibi

Alper Bayrak[1], Müzeyyen Sarıtaş[2]

[1]İzmir Yüksek Teknoloji Enstitüsü, Elek-Elektronik Mühendisliği Bölümü, 35430 Urla-İzmir
e-mail: alperbayrak@iyte.edu.tr
[2]Gazi Universitesi,Elektrik Elektronik Mühendisliği Bölümü, 06570 Ankara
e-mail: muzeyyen@gazi.edu.tr

Özet

Bu çalışmada, beş eksenli bir robot kolunun ters kinematik hesaplamaları ve yörünge planlaması yapılmıştır. Robotun hareketi esnasında önüne çıkabilecek engeli/hedefi tanıyacak bir sistem elde edilmiş ve robot kolu hareketinin simülasyonu bilgisayar ortamında gerçekleştirilmiştir. "Delphi 7" programlama dili, hesaplamalar ve robotun üç boyutlu simülasyonu için kullanılmıştır. Görüntü işleme ve engel/hedef tanıma "MATLAB R2006a" da gerçekleştirildikten sonra sonuçlar, "Delphi 7"deki simülasyon programına aktarılmıştır.

Anahtar kelimeler: Beş eksenli robot kolu, simülasyon, engel/hedef takibi

Simulation of a Robot Arm With Five Axes and Obstacle/Target Detection

Abstract

In this study, the inverse kinematics calculations and trajectory planning of a robot arm with five axes has been done. A system that could detect any obstacle/target appeared on the way of robot arm during its motion has been obtained and the motion of robot arm has been simulated by a computer. "Delphi 7" programming language was used for the calculations and the 3-dimentional simulation of the robot arm. After the image processing and the obstacle/target detection were done using "MATLAB R2006a", the results were transferred to the program written in Delphi 7"

Key Words: Five axes robot arm, simulation, obstacle/target detection

1. Giriş

Robotik, Makine Mühendisliği, Elektrik ve Elektronik Mühendisliği ve Bilgisayar Mühendisliği disiplinlerinin ortak çalışma alanıdır. Robotlar bir yazılım aracılığı ile yönetilen ve yararlı bir amaç için iş ve değer üreten karmaşık makinelerdir.

1.1. Robotların Kinematik Problemleri [1,2].

Bir robot kolu, birbirine seri ya da paralel olarak bağlı eklemlerden oluşur. Robot kolunun her bir eklemine, belirli açılar verilerek, robotun uç noktasının gideceği konumu belirlemek için kullanılan problemlere ileri kinematik problemler denir. Bunun tam tersi olarak, robot kolunun uç noktasının, istenilen yere gitmesi için eklemlerin alması gereken açıların bulunmasında kullanılan problemlere de ters kinematik problemler denir.

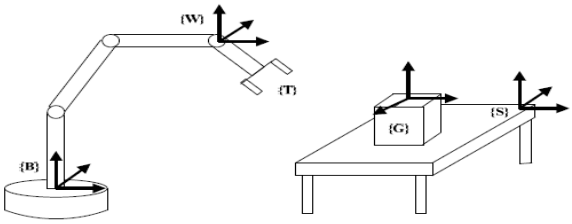
İleri Kinematik

Robotun ileri kinematiği, birbirine bağlı her kol çifti için, uç noktanın konfigürasyonunu gösterir. Robot kolu ve çevresindeki birimler, Şekil 1'de verilmiştir. Robotun sabit, yani hareket etmeyen parçasına, Ana Çerçeve {B} denir. İstasyon Çerçevesine {S}, evrensel çerçeve de denir ve robot bütün hareketlerini bu çerçevede yapar. Bilek Çerçevesi {W}, robotun son bağlantısına yerleştirilmiştir. Robotun işlem yapacağı nesnenin üzerindeki çerçeve ise, Hedef Çerçevesi {G}'dir.

Araç çerçevesinin, ana çerçeveye göre yönelimini veren ileri kinematik eşitliği ($g_{st}(\theta)$, aşağıdaki gibi elde edilir.

$$g_{st}(\theta) = e^{\xi_1\theta_1}e^{\xi_2\theta_2}.....e^{\xi_n\theta_{n1}}g_{st}(0) \qquad (1)$$

$e^{\xi\theta}$: her bir eklem için dönüşüm matrisi; θ: eklem açısı; ξ: bir eklemin büklüm eksenidir. ($g_{st}(0)$: robot manipülatörünün başlangıç konfigürasyonudur.



Şekil 1. Robot kolu ve çevre birimleri [1,3]

Ters Kinematik

Genellikle ters kinematik çözümler iki sınıfa ayrılabilir: Kapalı form çözümleri ve sayısal çözümler. Kapalı form çözümleri, istenen bir uç nokta konfigürasyonu için hızlı ve etkin bir hesaplama sağlar. Sayısal çözümler için etkileşimli bir işlem sırası uygular. Bu işlemler, bir küme doğrusal olmayan aritmetik eşitliği, geometrik ve aritmetik özellikler kullanarak çözmek içindir. Bu çalışmada, ters kinematik problemler, Paden-Kahan alt problemlerine indirgenerek çözülmüştür[2].

1.2. Yörünge Planlaması [1,2].
Bir uç işlevcinin başlangıç pozisyonu ve yönelimi ile hedef pozisyonu ve yönelimi verildiğinde, uç işlevcinin başlangıçtan hedefe doğru sarsıntısız bir yörüngede hareket etmesi istenir. Bu işlem, istenilen uç işlevci hızını ve dolayısıyla istenilen eklem hızlarını hesaplayarak yapılabilir[1-4].

Yörünge planlaması, noktadan noktaya (point-to-point) hareket ve sürekli yol hareketi olmak üzere ikiye ayrılır. Bu çalışmada, yörünge planlaması yapılırken, sürekli yol hareketi kullanılmıştır. Hareket esnasında pozisyonda, hızda ve ivmede sürekliliği sağlamak için beşinci derece polinomlardan yararlanılmıştır[1,2].

2. Robot Kolu Simülasyonu

Bu çalışmada, robot kolunun zaman içinde yaptığı hareketler Delphi 7.0 programlama dili kullanılarak, bilgisayar ortamında taklit edilmiştir. Simülasyonda, bilgisayar ekranında 3-boyut hissi uyandıracak şekilde görsel bir ortam hazırlanarak robotun hareketleri incelenmiştir.

Robot kolu simülasyonunda, kolun her bir parçası, kendinden önceki parça ile bir bütünlük içinde, fakat kendi içinde ayrı bir parça olarak hareket edecek şekilde tasarlanmıştır. Kol parçaları aynı özellikleri taşıdıkları için, programlama dilinin nesne tabanlı olması özelliği kullanılarak bir "robot_kolu" sınıfı oluşturulmuş ve her bir kol bu sınıftan nesne olarak türetilmiştir. Bu sınıfta, temel olarak her kolun uzunluğu, başlangıç ve bitiş noktaları ve eklem açısı gibi özellikleri bulunmaktadır.

Her bir kol parçası için değerler girildikten sonra kolların hareketini sağlamak için hareket fonksiyonları hazırlanmıştır[1,2]. Bu fonksiyonlar, robotun ileri kinematik problemlerini kullanarak verilen açı değerleri için her bir kol parçasının hangi konumu alacağını hesaplamaktadır. Her bir hesaplamadan sonra, kol nesnelerinin konum ve açı değerleri otomatik değiştirilmektedir.

Daha sonra robot kolunun, ekranda 3-boyutlu olarak çizdirilmesi için, çizim fonksiyonları hazırlanmıştır[1].

Robot_kolu sınıfı
Robot kolunun parçalarını oluşturmak için, "robot_kolu" adında bir sınıf oluşturulmuş ve robot kolunun manipülatörlerini simgeleyen parçalar bu sınıftan üretilen bir dizinin elemanları olarak türetilmiştir. Bu sınıfta, bir parçayı tanımlamak için gerekli olan uzunluk, başlangıç ve bitiş koordinatları ve açı gibi özellikler belirtilmiştir.

Hareket fonksiyonları

Robot kolunda beş adet hareketli parça bulunmaktadır. Bu parçaların açıları değiştiğinde uç noktanın alacağı değer ileri kinematik problemi ile bulunabilir. Her parça kendisinden önceki parçaları ve kendisini problemin parametreleri olarak kabul eder. Aşağıda, birinciden dördüncüye kadar her parçanın uç noktasının hesaplanmasında ve simülasyonunda kullanılan problemler verilmiştir.

$$g_{st}(\theta) = e^{\hat{\xi}_1\theta_1} g_{st}(0) \quad \text{.....} \quad (2)$$

$$g_{st}(\theta) = e^{\hat{\xi}_1\theta_1} e^{\hat{\xi}_2\theta_2} g_{st}(0) \quad \text{.....} \quad (3)$$

$$g_{st}(\theta) = e^{\hat{\xi}_1\theta_1} e^{\hat{\xi}_2\theta_2} e^{\hat{\xi}_n\theta_{n1}} g_{st}(0) \quad \text{..... ...} \quad (4)$$

$$g_{st}(\theta) = e^{\hat{\xi}_1\theta_1} e^{\hat{\xi}_2\theta_2} e^{\hat{\xi}_3\theta_3} e^{\hat{\xi}_4\theta_4} g_{st}(0) \quad \text{.....} \quad (5)$$

Bu problemler kullanılarak her bir parçanın uç noktasının koordinatları hesaplanır. Burada 5 no'lu eklem parçası (uç nokta) için denklem yazılmamıştır; bunun nedeni uç noktanın konumunu değiştirmiyor olmasıdır. Bu nedenle beş numaralı parçanın, dört numaralı parçanın uzantısı olarak, simülasyonu yapılmıştır.

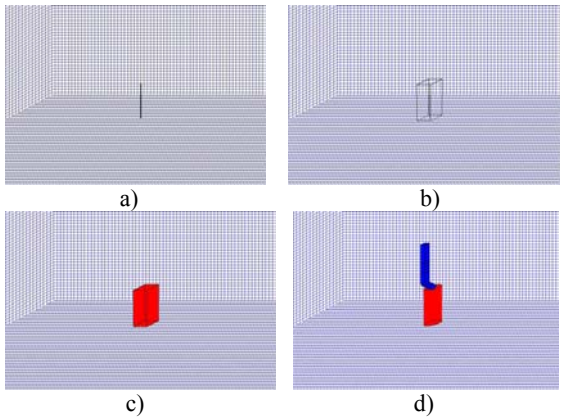
Çizim fonksiyonları:
"Sekil_ciz" *fonksiyonu:* Simülasyonda öncelikle parçaların merkezinden geçen doğru parçaları çizilir. Bunun için "Se*kil_ciz*" fonksiyonuna sırayla parçaların uç ve dip noktalarının x,y,z koordinatları gönderilir. Görüntü üzerinde bu üç boyutun iki boyuta iz düşümleri çizileceği için gönderilen bu koordinatlar iki boyuta indirgenir ve Şekil 2-a'daki gibi bir çizgi elde edilir.

$$x' = x - y\cos(gea) \quad (6)$$

$$y' = z - y\sin(gea) \quad (7)$$

Burada, gea: görüntü açısı=30^{o}dir.

"Prizma_ciz" fonksiyonu: Çizgi çizildikten sonra etrafına, bu çizgiyi merkez kabul edecek şekilde bir dikdörtgenler prizması, üç boyut hissi vermesi için yerleştirilir. Bu prizma, çizgi ile hareket edecek şekilde tasarlanmıştır. Prizma çizildikten sonra Şekil 2-b'deki görüntü elde edilmiştir.



Şekil 2. Adım adım prizma çizimi a) "Sekil_ciz" fonksiyonu ile elde edilen tek çizgi b) Tek çizgi etrafına giydirilmiş prizma görüntüsü c) Boyanmış prizma görüntüsü d) 3-boyutlu robot kolu görüntüsü

"Boya" fonksiyonu: Prizmanın 12 kenarı bulunmaktadır. Bu kenarlardan karşılıklı olanların araları boyanarak hacim elde edilir (Şekil 2-c). Tek bir parça çizildikten sonra diğer parçalar da çizilerek 3-boyutlu robot kolu görüntüsü oluşturulmuştur(Şekil 2-d).

"*Simülasyon*" fonksiyonu, yukarıdaki tüm hareket ve çizim fonksiyonlarını belirli bir sırayla çalıştırarak hareketli görüntüyü veren bir fonksiyondur.

3. Görüntü İşleme [5-9] ile Engel/Hedef Nesne Takibi[1]

Engel/Hedef nesne tanımlama:
Bu çalışmada, engel veya hedef nesnenin bilgisayar tarafından, girilen görüntü içinden bulunması istenmektedir. Görüntü içinde bulunması istenen hedef nesne birçok farklı boyutta olabilir. Bu yüzden bilgisayara, hedef nesnenin farklı boyutlardaki görüntüleri önceden tanıtılmalıdır.

Burada, hedef nesne tanımlama işlemi için seçilen nesnenin fotoğrafları, daha önceden çekilmiş fotoğraflar içinden kesilerek çıkartılmış ve bir klasör içine kaydedilmiştir. Uygulamamızda nesnenin hareket alanı dar olduğu için dört adet örnek yeterli bulunmuştur.

Hedef nesne tanımlama yöntemi:
Önce, 4 adet hedef nesne içeren bir veri tabanı oluşturulmuştur. Veri tabanına kaydedilen hedef nesne görüntüleri (Şekil 3), ortam fotoğraflarından kesilerek çıkartılmış ve ikili kodlu görüntüye dönüştürülerek .jpg uzantısı ile kaydedilmiştir

Şekil 3. Veri tabanındaki hedef nesne görüntülerinden biri

MATLAB ortamında her bir hedef nesne görüntüsü, ait olduğu ortam görüntüsü üzerinden maske olarak geçirilmiştir (Şekil 4) [1, 8]. Katlama işlemi şu şekilde gerçekleştirilmiştir:

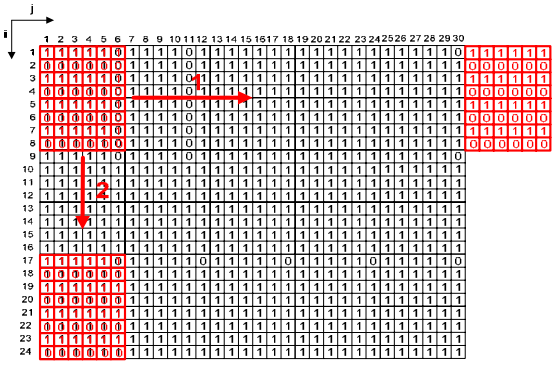
i) Ortam görüntüsünün boyutları, ixj (600 satırx800 sütün) olarak tanımlanmıştır. Her bir hedef nesne (maske) görüntüsü ise, boyutlarına göre nxm şeklinde tanımlanmış olup en az 330 satır ve 130 sütundan oluşmaktadır ve maske boyutu dinamik olarak değişmektedir.

ii) Katlama işlemi, maskenin 1. satırı, ortam görüntüsünün 1. satırına gelecek şekilde başlatılmış ve satır boyunca birer sütün atlanarak sağa doğru kaydırılmıştır.

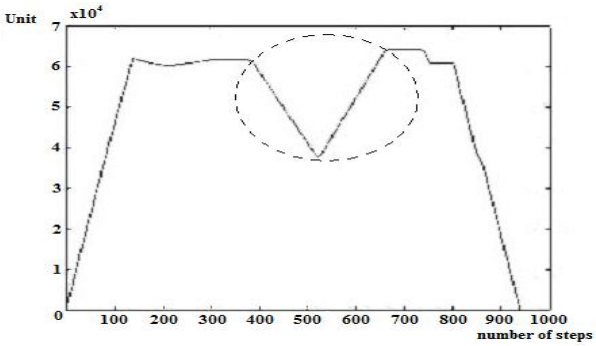
iii) Katlama işleminde maske, her bir satır işlemi bittiğinde bir alt satıra kaydırılmış ii)'deki işlemler tekrar edilmiştir. Maskenin en son satırı, görüntünün en son satırını geçtiğinde ise tarama işlemi sonlandırılmıştır. Bu çalışmada işlemin daha kısa sürmesi için maske dörder satır atlatılarak (i=1-5-9…) tarama işlemi tekrar edilmiştir.

iv) Katlama işlemi gerçekleştirilirken; maskenin her bir sütunu, bu sütünün altında kalan görüntü sütunundaki piksel değerleri ile toplanmış ve sütun toplamı olarak alınmıştır. Bu işlem, maskenin her bir sütunu için ayrı ayrı tekrarlanmış ve maskenin tüm sütunların toplamı olarak bir matriste kaydedilmiştir. Bundan sonra maske, görüntü üzerinde bir sütun kaydırılarak aynı işlem yapılmış ve maskenin birinci sütünü ortam görüntüsünün son sütununu geçinceye kadar tekrarlanmıştır. Böylece, bu işlem, j+m-1 adımda tamamlanmıştır. Her adımda, maske ve maskenin altındaki piksellerin toplamı (en fazla nxm şeklinde) alınmış, ve birim olarak başka bir matrise (1xk) kaydedilmiştir.

Maskenin, görüntüdeki nesnenin üzerinden geçerken, elde edilen grafiği Şekil 5'te verilmiştir. Bu şekilde; x ekseni(toplam piksel), j+m-1 adımına; y ekseni (adım sayısı)ise, maske (nxm) ve altındaki pikseller toplamına karşılık gelmektedir..



Şekil 4. Hedef nesnenin(maske), resim üzerinde örnek katlama işleminin görüntüsü



Şekil 5. Katlama sonucunda, maskenin, resimdeki hedef nesnenin üzerinden geçerken, elde edilen grafiği

Bu grafikte görünen V şeklindeki kısmın başlangıç noktası, ortam görüntüsündeki hedef nesnenin başlangıç noktasına denk gelmektedir.

Bilgisayarda, bu grafiksel şeklin algılanabilmesi ve diğer benzerlerinden ayırt edilebilmesi amacıyla, V şeklindeki kısım için çeşitli karakteristik parametreler tanımlanmış ve aşağıda verilmiştir.

- **Eğim 1 (e1):** İnen kenarın eğimi, **Eğim farkı 1 (ef1):** İnen kenarın doğrusallığı, **Genlik 1 (g1):** İnen kenarın genliği, **Eğim 2 (e2):** Çıkan kenarın eğimi, **Eğim farkı 2 (ef2):** Çıkan

kenarın doğrusallığı, **Genlik 2 (g2):** Çıkan kenarın genliği, **Keskinlik (k):** İki kenar arasındaki açı, **Genlik oranı (go):** İki kenarın genlik oranı.

Hedef nesnenin, görüntüde bulunduğu yerdeki bu değerler, önceden hesaplanmış ve o nesne görüntüsü için referans değer olarak kullanılmıştır.
Referans değere en yakın uzaklığı (Eş. 8) veren satırın uzaklık değeri, belli bir değerin altında ise nesnenin bulunduğu anlaşılır aksi halde nesnenin bulunamadığı varsayılır ve bir sonraki nesne görüntüsü kullanılarak aynı işlemler tekrarlanır.

Uzaklık=|(e1-re1)|+|(ef1-ref1)|+|(g1-rg1)|+|(e2-re2)| +|(ef2-ref2)|+|(g2-rg2)|+|(k-rk)|+|(go-rgo)| .... (8)

Hedef nesne tanımlama algoritmasının blok diyagramı, Şekil 6'da sunulmuştur.

Yukarıda anlatılan işlemler, Şekil 7'deki gibi arka fona sahip ve sadece bir hedef nesnenin bulunduğu bir görüntü üzerinde hedef nesnenin tespit edilmesini açıklamaktadır. Ancak, Şekil 8'deki gibi arka fonun farklı olduğu ve tespit edilmek istenen nesne dışında nesnelerin de bulunduğu bir görüntüde, istenen hedef nesnenin tespiti zorlaşmaktadır.

Bu durumda algoritma, yanlış bir yeri veya bulunması istenen nesne ile birlikte etrafındaki benzer nesneleri de işaretleyebilmektedir.

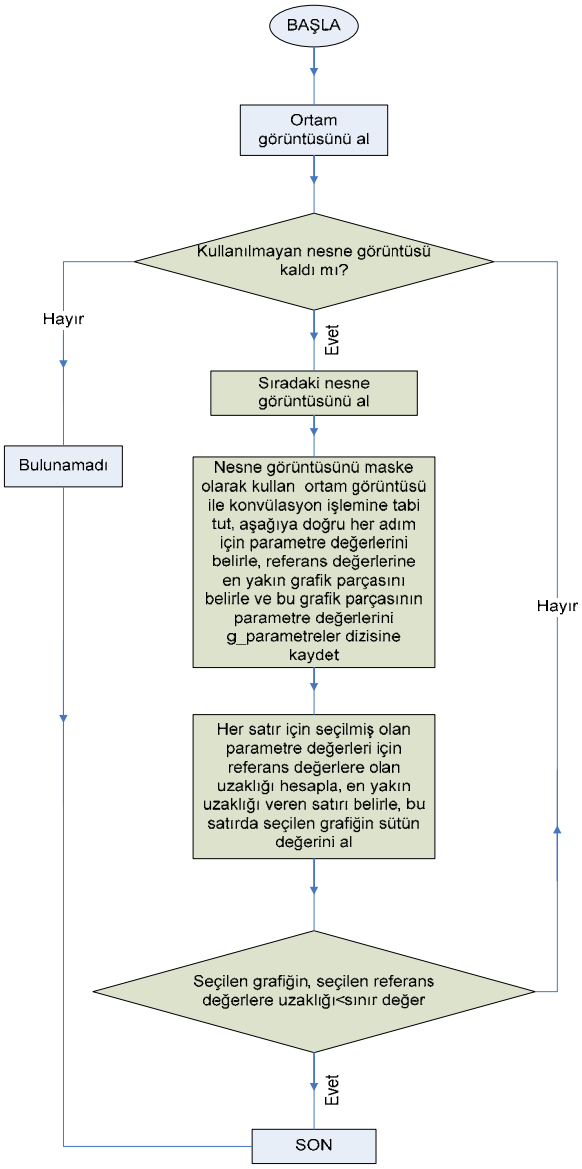
Bu sorunu çözmek için, seçilen bölge görüntüden çıkartılır. Elde edilen bu görüntü üzerinde gürültüyü azaltmak için erozyon işlemi uygulanır (Şekil 9).

Daha sonra, etiketleme işlemi ile Şekil 9'da görülen görüntü üzerindeki, birbirinden bağımsız beyaz bölgeler numaralandırılır. Numaralandırılan bu bölgelerin içerdikleri piksel sayısı o bölgenin ağırlığını verir. Ağırlığı 3000'den küçük olan bölgeler silinir. Görüntü içinde aradığımız hedef nesne üç ayrı beyaz bölgeden oluşmaktadır ve bu bölgelerin yatay kenarları dikey kenarlarına göre daha uzundur. Buradan yola çıkarak görüntü içinde yatay kenarları dikey kenarlarından küçük olan bölgeler de silinir.

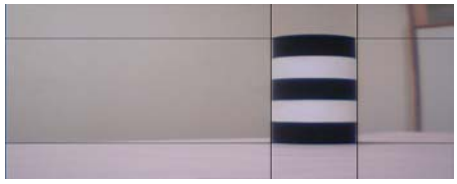
Görüntüyü temizleme işlemi bittikten sonra, geriye kalan bölgelerin ağırlıklarının ortalaması alınır. Bölgeler, bu ortalamaya olan uzaklıklarına göre sıralanır. İlk üç bölge, nesneye ait bölgeler olarak kabul edilir. Ancak bu bölgelerin ağırlıkları arasında da bir oran olması gerekir. Bu oran deneysel olarak en fazla 1.7 olarak belirlenmiştir. Oran tutuyorsa nesnenin bulunduğu anlaşılır. Son olarak, bu üç bölgenin sınırlarına bakılarak nesnenin yatay ve düşey konumları hesaplanır. Hesaplanan konum değerleri ortam görüntüsü üzerinde işaretlenir(Şekil 10).

Eğer yukarıdaki işlemlerin sonucunda istenilen şartlara uygun bölgeler tespit edilememişse, bu bölgede nesnenin olmadığı yani yanlış bir bölgenin seçilmiş olduğu anlaşılır. Ana programa dönülür. Gerçek görüntüde, seçilen bu bölge çıkartılır yani beyaza boyanır. Yeni oluşan görüntü tekrar taranır.

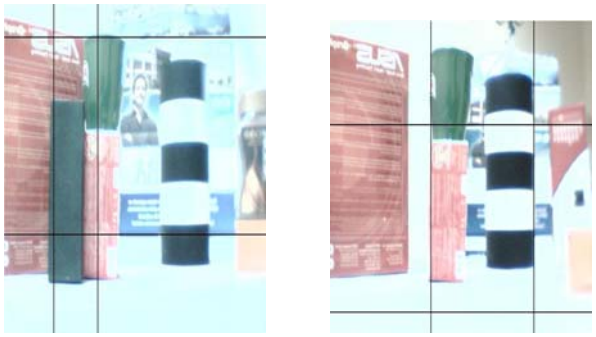
Uygulamada, Şekil 7'deki gibi arka fonun boş olduğu ortam görüntüsünde, istenen hedef nesnenin bulunması 198.95 sn, Şekil 10-a'daki gibi farklı bir arka fona sahip ve bulunması istenen hedef nesne dışındaki nesnelerin bulunduğu ortam görüntüsünde, hedef nesnenin bulunması 199.16 sn, Şekil 10-b'deki gibi bulunması istenen nesneye benzer bir nesnenin de yer aldığı ortam görüntüsünde, istenen hedef nesnenin bulunması 202.34 sn zaman almıştır.
Ortam görüntüsündeki benzer hedef nesne sayısı arttıkça, istenen hedef nesnenin bulunma olasılığının azaldığı ve işlem süresinin de arttığı gözlenmiştir.



Şekil 6. Hedef nesne tanıma algoritmasının iş akış şeması



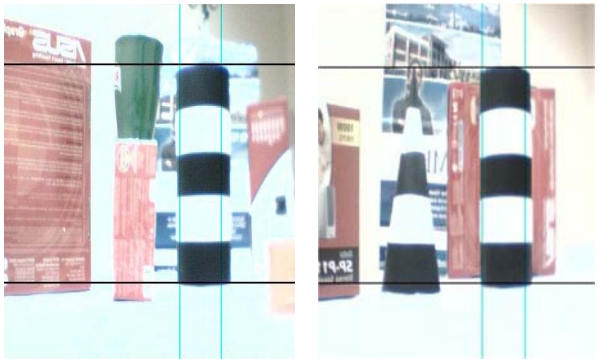
Şekil 7. Hedef nesnenin tespit edilmesi halinde, konumunun görüntü üzerinde işaretlenmesi



a) b)

Şekil 8. Görüntüde hedef nesnenin yanlış işaretlendiği bölgeler

Şekil 9. Erozyon uygulanmış hedef nesne görüntü



a) b)

Şekil 10. a) Hedef nesnenin yerinin tespit edildiği görüntü, b) Aranan hedef nesneye benzer başka bir nesnenin bulunduğu ortamda, hedef nesnenin tespit edildiği görüntü.

4. Koordinat Tespiti

Kamera ile elde edilen görüntülerde her bir piksel, aynı zamanda mesafe bilgisi içerir. Hedef nesne ile kamera arasındaki uzaklığı cm cinsinden veren formül ölçümler sonucunda aşağıdaki gibi bulunmuştur.

$$|HGx| = 0.0151*\left(\frac{pu}{10}\right)^3 - 0.03122*\left(\frac{pu}{10}\right)^2 + 2.0494*\left(\frac{pu}{10}\right) - 32.4178 \quad ....(9)$$

pu: piksel uzaklığı, |**HGx**|: hedef nesne ve kamera arasındaki, cm cinsinden uzaklık

5. Sonuçlar

Bu çalışmada, önce, beş eksenli dört eklemli bir robot kolunun ters kinematik hesaplamaları ve yörünge planlaması yapılmış, robot kolunun hareketi esnasında yoluna çıkabilecek engelleri/hedefleri algılayabilmesi için bir kamera ve bilgisayar programından oluşan bir sistem geliştirilmiş ve robot kolunun hareketinin simülasyonu bilgisayar ortamında gerçekleştirilmiştir.

Hazırlanan simülasyon programı, eklem açılarını 25 msn'lik hassasiyetle hareket ettirebilmektedir. Gerçek zamanlı robot hareketi, 0.5-2 sn arasında zaman almaktadır. Bilgisayarda, işlem süresinin uzun olmasından dolayı, bu bilgisayarın hızına da bağlı olarak, robot kolu hareketi 5-18 sn arasında gerçekleşmektedir

Bu çalışmada, kullanılan hedef nesne tanımlama algoritması laboratuar ortamında %90'nın üstünde bir başarı sergilemiş ise de, hedefi tespit etme süresi açısından yeterince verimli değildir. Görüntüdeki, hedef nesnenin algılanma süresi en az 1.5 dakika almaktadır. Bu süre, engelin/hedefin sürekli bir şekilde takip edilmesi imkanını azaltmaktadır. İleride, bu algoritmada bazı değişikler yapılarak veya farklı algoritmalar kullanılarak bu süre kısaltılabilir ve böylece çalışmanın uygulama alanı genişletilebilir.

Kaynaklar

1. Bayrak A., "Beş Eksenli bir Robot Kolunun Simülasyonu ve Kontrolü", Yüksek Lisans Tezi, ***Gazi Üniversitesi, FBE***, Ankara, 2007
2. Tonbul T.S., "Beş Eksenlı Bir Robot Kolunun Ters Kinematik Hesaplamalarının ve Yörünge Planlamasının Yapılması", Yüksek Lisans Tezi, ***Gazi Üniversitesi FBE,*** Ankara, 2002.
3. Bingül Z., Küçük, S., Robot Tekniği-I, ***Birsen Yayınevi***, İstanbul, 1-17 23-59 (2005).
4. İnternet: Kumar, V., "Cartesian Trajectory Planning for Robot Manipulators", University of Pennsylvania, School of Engineering and Applied Science Department of Mechanical Engineering. http://www.seas.upenn.edu/~meam520/notes02/CartesianControl10.pdf, (2006).
5. Kert M., "Gerçek Görüntüden Elde Edilen Koordinatlarla Robot Kol Hareket Optimizasyonu", Yüksek Lisans Tezi, ***Mustafa Kemal Üniversitesi, Fen Bilimleri Enstitüsü***-Antakya, 13-44 (2006).
6. Robin N. Strickland, **"**Wavelet Transform Methods for Object Detection and Recovery", **IEEE Transactions On Image Processing**, 6(5):724-735 (1997).
7. Ostermann, L. G., Clairon, P. S., "Hierarchical Region Based Processing of Images and Video Sequences: Application to Filtering, Segmentation and Information Retrieval", Doktora Tezi, ***Universitat politècnica de Catalunya, Signal Theory and Communications Department***, 1-191 (2002).
8. Gonzalez R. C., "Digital İmage Processing", **Prentice Hall**, 147-278, (2002).
9. Gonzalez R. C., "Digital Image Processing Using MATLAB",***PEARSON Prentice Hall,*** 359-361,345-346 (2004).