Uluslararası Teknolojik Bilimler Dergisi
International Journal of Technological Sciences
UTBD
IJTS

Araştırma Makalesi/Research Article

Gerçek zamanlı işletim sistemi ile çalışan sumo robot

Ahmet Akif Köse[1], Mehmet Albayrak[*2]

[1]Isparta Uygulamalı Bilimler Üniversitesi, Uzaktan Eğitim Meslek Yüksekokulu, Bilgisayar Teknolojileri Bölümü, 32260, Isparta, Türkiye
[2]Süleyman Demirel Üniversitesi, Fen Bilimleri Enstitüsü, Elektronik Bilgisayar Eğitimi Bölümü, 32260, Isparta, Türkiye

Anahtar Kelimeler
Sumo robot;
Gerçek zamanlı
İşletim sistemi;
Kontrol.

Makale geçmişi:
Geliş Tarihi: 30.07.2019
Kabul Tarihi: 03.05.2020

Özet: Bu çalışmada; tasarımı ve imalatı yapılan sumo robotun, gerçek zamanlı işletim sistemi kütüphanesi kullanılarak karar vermesi ve mümkün olan en hızlı şekilde komutu uygulaması ve böylece rakiplerine karşı üstünlük sağlaması amaçlanmıştır. Mevcut sumo robot uygulamalarında geçerli ölçü olan 20cm ×20cm boyutlarına sığabilecek dört tekerli bir robot mekaniği hazırlanmıştır. Bu mekanikte kullanılan motorlar 1000d/dk üzerinde devire sahip olması ve yüksek güç üretmesi planlanmıştır. Bunların yanı sıra kullanılan kontrast sensörleri tepki süresi 1ms'den az olacak şekilde seçilmiştir. Kontrol kartında kullanılan mikrodenetleyici/mikroişlemci seçiminde, gerçek zamanlı işletim sistemi komutlarını destekleyecek ve mümkün olan en yüksek frekansta çalışabilecek işlemci tercih edilmiştir. Hızlı algılama ve hızlı karar verme sonucunda yüksek olan motor devirleri de dikkate alınırsa hızlı hareket (çabuk tepki verme) sağlanmıştır. Bu çalışmayı benzer çalışmalardan ayıran ana unsur, seçilen donanımsal parçaların yüksek performansa (hız, güç, dayanıklılık vb.) sahip olması ve gerçek zamanlı işletim sistemi (GZİS) komutları ile yazılımsal olarak üstünlük elde edilmesi amaçlanmıştır. Fakat seçilen donanımsal bileşenler yüksek performanslı çalışmaya imkân tanımış olsa da, kullanılan mikroişlemcinin tek çekirdekli olması sebebi ile iş parçacıklarını istenen performansta çalıştıramaması çalışma başlangıcında hedeflenen performans kadar iyi sonuçların elde edilememesine neden olmuştur.

Atıf için/To Cite: Köse AA. Albayrak M. Gerçek zamanlı işletim sistemi ile çalışan sumo robot. Uluslararası Teknolojik Bilimler Dergisi, 12(1), 27-33, 2020.

Sumo robot working with real-time operating system

Keywords
Sumo robot;
Real-time
Operating system;
Control.

Article history:
Received: 30.07.2019
Accepted: 03.05.2020

Abstract: In this study; sumo robot, designed and manufactured, is intended to make decisions using the real-time operating system library and to execute the command as quickly as possible, thereby gaining superiority over its competitors. A four-wheeled robot mechanics has been prepared which can fit "20cm×20cm" which is the current dimension in current sumo robot applications. The motors used in this mechanic are planned to have speeds above 1000rpm and to produce high power. In addition, the contrast sensors used were selected to have a response time of less than 1ms. In the selection of the microcontroller/microprocessor used in the control board, the processor that supports real-time operating system commands and can operate at the highest possible frequency is preferred. Rapid movement (quick response) is provided if high engine speeds are also taken into account as a result of rapid detection and quick decision making. The main factor that distinguishes this study from similar studies is that the selected hardware parts have high performance (speed, power, durability, etc.) and software superiority with real-time operating system (RTOS) commands is aimed. However, although the selected hardware components allowed high performance operation, the fact that the microprocessor used was single core and the inability to run the threads at the desired performance caused the results not to be as good as the targeted performance at the beginning of the study.

* İlgili yazar/Corresponding author: mehmetalbayrak@isparta.edu.tr

1. Giriş

Robot, otonom veya önceden programlanmış görevleri yerine getirebilen elektro-mekanik bir cihazdır. Robotlar doğrudan bir operatörün kontrolünde çalışabildikleri gibi bağımsız olarak bir bilgisayar programının kontrolünde de çalışabilirler. Katipoğlu çalışmasında; robot deyince insan benzeri makineler akla gelse de robotların çok azı insana benzer şeklinde nitelendirmiştir (1). Robot kelimesi; ilk olarak Karel Capek' in 1920 yılında yazdığı "R.U.R. (Rossum' s Universal Robots)" adlı eserinde yer almıştır (2). Robotik alanındaki çalışmalar günümüzde; endüstriyel robotik, operasyonel robotik, tıp ve sağlıkta robotik, sibernetik, oyuncak robotlar ve hobi amaçlı robotlar olarak kategorize edilmektedir. Robotik teknolojisinin uygulamada geldiği en son nokta için, Boston Dynamics' in Atlas isimli robotu örnek verilebilir (3).

Bu çalışmada; hobi amaçlı robotlar kategorisinde bulunan sumo robot üzerinde bir uygulama geliştirilmiştir. Sumo robotların tarihsel gelişimine bakıldığında, insanların hobi ve oyun amaçlı küçük ölçekte pek çok robot yaptığı görülmektedir (4). Sumo robotlar dünya çapında çok ilgi görmekte ve sumo robotlarla ilgili ulusal ve uluslararası pek çok turnuva düzenlenmektedir. Bu turnuvalar 1980' li yılların sonunda ortaya çıkmıştır. Özellikle Amerika ve Kanada da büyük ilgi görmüştür. Sumo robot ilk olarak 1980' lerin sonunda, Japonya Fuji yazılım şirketinin başkanı Hiroshi Nozawa tarafından icat edilmiştir. İlk gösteri oyunu Ağustos 1989' da gerçekleştirilmiş ve ilk resmi turnuva 1990 yılında yapılmıştır.

Sumo robotlarını rakiplerine karşı başarılı kılan; mekanik kısımlarının güçlü ve sağlam oluşundan ziyade, sensörlerin algılama hızının yüksek olması ve işletim sisteminin kararlı çalışmasıdır. Bu çalışmada; donanımsal bileşenlerin yüksek performanslı olmasının yanı sıra, robotun işletim sisteminin de gerçek zamanlı işletim sistemi olmasının pozitif bir katkı sağlayacağı düşünülerek yola çıkılmıştır. Gerçek zamanlı çalışan sumo robot, adından da anlaşıldığı üzere gerçek zamanlı bir işletim sistemi desteği ile çalışmaktadır. Bu durum; diğer sumo robotlarla karşılaştırıldığında, gerçek zamanlı çalışan sumo robotu ayrıcalıklı kılan en önemli özelliklerinden birisidir. Bu özellik sayesinde, yazılımsal olarak yerine getirdiği görevleri daha hızlı yapabilmesi ve görevlerin öncelik sırasını belirleyebilme amaçlanmıştır.

Bu çalışmada hedeflenen, gerçek zamanlı işletim sistemi ile çalışan sumo robot ile gerçek zamanlı işletim sistemi ile çalışmayan sumo robot arasındaki farkları deneysel olarak ortaya koymak ve bir diğer hedef de hangi şartlarda üstünlük sağlayabileceğini diğer benzer çalışmalarda olduğu gibi deneysel olarak ispatlamaktır. Çalışmanın diğer çalışmalardan farklı bir yönü çok çekirdekli yapılarda GZİS kullanımının üstünlüğü net olarak bilinmesine rağmen, bu çalışmada seçilen işlemci tek çekirdekli yapıya sahiptir. Bu şartlar altında yüksek performans elde edilmesi denenmiştir.

Sumo robotlar ile yapılan başka bir çalışmada tekerlekli mobil robot için tek kartlı bilgisayarda gerçek zamanlı işletim sistemi (GZİS/RTOS) performansını analiz etmeye çalışılmıştır. GZİS, genel amaçlı işletim sisteminden (GAİS/GPOS) daha iyi sistem yanıtına sahiptir. Bu sistem yanıtı mobil robot için önemlidir. Örneğin mobil robot, sensör duvarı algıladığında mümkün olan en kısa sürede durmalıdır. Kontrol kartı üzerinde GZİS sistem analizi, sensörün algıladığı mesafe ölçüm hassasiyeti, ortaya çıkan gecikme değeri ve ortamdaki sistem etkileşimi nedeniyle veri yenileme süreci gibi çeşitli noktaları kapsar. Elde edilen sonuçlar, çoklu iş parçacığı (multithread) aracılığı ile robotun 1 saniyeden daha kısa sürede yanıt verebilme yeteneğini kanıtlanmış ve kontrol kartına uygulanmıştır. Bu sonuç esnek gerçek zamanlı sistem olarak açıklanabilir. GZİS ve GAİS performansları kıyaslandığında, her iki sistem arasındaki ortalama gecikme farkının GZİS kullanıldığında 300ms kadar daha iyi olduğunu göstermiştir (5).

Bu konudaki başka bir önemli çalışmada; farklı mikroişlemciler için GZİS/GAİS birlikte çalıştıran çoklu işletim sistemi mimarisi uygulanmıştır. Bu mimarinin kullanımı ile GAİS tarafında değişiklik yapılması gerekmemektedir. Bu mimari ile çok daha az mühendislik maliyeti ile yüksek performans elde edebilmektedir. Çoklu işlemci üzerinde çalışan bu çoklu işletim sistemi mimarisi RGMP olarak adlandırılmıştır. Çoklu işletim sistemi kullanımında; GAİS olarak Linux ve GZİS olarak µC/OS-II kullanılarak uygulanması daha verimli sonuçlar vermiştir. Sonuçta oluşturulan yeni mimarinin gerçek zamanlı çalışan ve Linux uygulama arayüzü kullanan mimariden daha hızlı olduğu ve kullanılan iki ayrı işletim sisteminin birbirini etkilemediği görülmüştür (6).

2. Materyal ve Metot

Çalışma kapsamında tasarımı ve imalatı gerçekleştirilen robot, bir sumo robottur. Üzerinde çalışan işletim sistemi bakımından gerçek zamanlı işletim sistemi (GZİS) tercih edilmiştir. GZİS' nin tercih edilme sebepleri ilerleyen bölümlerde açıklanmıştır.

GZİS ile çalışan sumo robotu oluşturan bileşenler, üç ana başlık altında ele alınmıştır. Bunlar; mekanik, elektronik ve yazılım bölümleridir. Bir sumo robotun tasarımındaki başarı hedefi, rakibini en kısa sürede dohyonun dışına atmak olduğundan, mekaniğinin (gövde) sağlam ve dayanıklı olması önemlidir. Mekanik

aksamda bir diğer dikkat edilmesi gereken konu ise motorlar ve tekerleklerin seçimidir. Motorlar seçilirken; yüksek devirli ve yüksek torka sahip olmasına dikkat edilmelidir, böylelikle rakiplerine karşı üstünlük sağlayabilir. Elektronik devre elemanları kısmında ise; kontrol kartı, sensörler ve batarya seçimi önemlidir. Sumo robotu oluşturan bileşenlerden sonuncusu da yazılımdır.

Gerçek zamanlı işletim sistemi ile çalışan sumo robot; mekanik bileşenlerini oluştururken, yüksek tork ve devire sahip motorlar tercih edilmiştir. Bu aşamada, motor ve redüktöre uyumlu bileşenler seçilmiştir. Sonraki aşamada, tekerlekler alüminyum malzemeden üretilmiştir ve kaymasını azaltmak amacı ile silikon kaplanmıştır. Robotun mekaniği (gövdesi) için 2mm kalınlığında demir malzemeden üretilmiş lazer kesim gövde seçilmiştir.

Robotun elektronik bileşenleri; bataryalar, sensörler, motor sürücü devreleri ve kontrol kartıdır. Bataryalar, robotun ihtiyacı olan gücü, uzun süre çalışacak şekilde sağlamalıdır. Bu şartı yerine getirirken; bataryaların boyutlarının, robotun ağırlık sınırı ve mekanik ölçüleri içerisinde olmalıdır. Bu durumda batarya; küçük boyut, anlık yüksek akım kapasitesi, düşük ağırlık özelliklerini desteklemesi önemlidir. Seçim yapılırken bu hususlar göz önünde tutularak, Lityum–Polimer (LiPo) bataryalar tercih edilmiştir. Bataryalar, akım kapasitesi olarak 2600mAh ve 11.1Volt olarak belirlenmiştir. Batarya seçiminde; LiPo türü tercih edilmesinin bir sebebi de anlık çok yüksek deşarj akımını (dört motorun yüklü durumda çektiği 4x12A=48A) sağlamaktadır. Bu durum standart 2700mAh şarjlı piller ile kolaylıkla sağlanamamaktadır.

Sensörler robotun hızlı tepki vermesinde rol oynayan önemli bileşenlerden biridir. Robotun elektronik yapısında iki tip sensör kullanılmıştır. Birincisi mesafe sensörleridir. Mesafe sensörlerinin seçiminde tepki süresi dikkate alınmıştır. Bu robotta kullanılan mesafe sensörlerinin tepki süresinin 1ms gibi hızlı bir değere sahip olması, rakipleri algılama adına üstünlük sağlamaktadır. Bunun yanı sıra, kızılötesi çalışma sisteminin pek çok sensörde yer aldığı bilinmektedir. Bu tür sensörlerde tepki (cevap) süresinin 5ms' nin üzerine olduğu görülmektedir. Robotta; rakibini algılaması için, yaklaşık 270 derecelik görüşü tek yönlü hareket ile sağlayan sensörler kullanılmıştır. Bu sensörler, 3 adedi önde, 2 adedi yanlarda olmak üzere toplam 5 adettir. Robotta kullanılan bir diğer sensör tipide kontrast sensörüdür. Kontrast sensörleri robotun çalışma alanı içerisinde kalmasını sağlamakta ve sadece önlerde olmak üzere 2 adet kullanılmıştır.

Elektronik bileşenlerden bir diğeri de motor sürücü devresidir. Motor sürücü devresi VNH2SP30 tipinde seçilmiştir. Bu motor sürücü devresinin tercih edilmesinde en önemli sebeplerden birisi motorların (sağ iki, sol iki) kilitli motor ve/veya tam yükte çalışması durumunda 2x12A=24A anlık akım çekilmesidir. Buna karşılık sağ ve sol yönler için motor sürücü devresinin 30A' lik değerinin bu akımı karşılayabilmesidir. Ayrıca, rakip algılandığında kullanılan değişken hız kontrolü için de (PWM kontrol) destekler.

Robotu oluşturan son elektronik bileşen kontrol kartıdır. Kontrol kartı olarak, Arduino Mega 2560 Kontrol Kartı tercih edilmiştir (7). Robotun yazılımsal bileşenleri bu çalışmada robotun en öne çıkan özelliğidir. Robot, hazırlanan yazılım sayesinde otonom olarak hareket edebilmekte ve değişen durumlara göre pozisyon alabilmektedir. Ayrıca yazılımda gerçek zaman desteği olması, robotun birçok işlevi hızlı ve aynı zamanda gerçekleştirilmesine de imkân vermektedir.

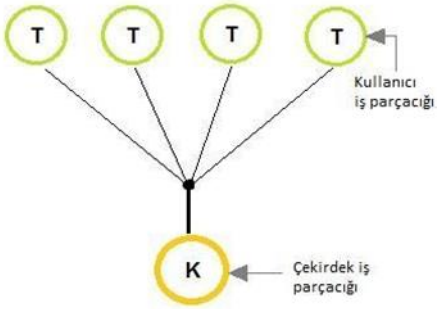
Gerçek zamanlı işletim sistemi (GZİS) kullanılarak, robotun çoklu görevleri hızlı ve doğru bir şekilde gerçekleştirebilmesi amaçlanmıştır. Gerçek zamanlı işletim sistemi günümüzde birçok farklı alan ve uygulamada kullanılmaktadır. GZİS belli bir zaman dilimi içerisinde, mantıksal olarak doğru sonuç elde edilmesi gereken gerçek zaman uygulamalarında ve gömülü sistemlerde sıklıkla kullanılır. GZİS zamana bağlı olarak kaynakları verimli şekilde kullanabilmektedir. Gerçek zamanlı işletim sistemi yapısal olarak; gerçek zaman ve işletim sistemi olmak üzere iki bölümden oluşmaktadır. Gerçek zaman kavramı; gerçekleştirilen işlemlerin yanıt verme süresinin belli bir değer aralığında olmasıdır. Gömülü sistemlerde bu değer aralığının önemi artmaktadır. Bu nedenle, değerin öneminin yüksek olduğu sistemlerde gerçek zaman olgusunun garanti altına almak için bazı mekanizmalar kullanılır. Bu pencereden bakıldığında GZİS gerçek zamanı garanti eden bir mekanizmadır. Gerçek zamanlı işletim sistemini oluşturan bölümlerden bir diğeri de işletim sistemi kısmıdır. İşletim sistemi geliştirilen uygulamalar ile donanım arasında bir katman sağlar. GZİS kullanıcıların donanımlara seçilen programlama dillerinin aracılığı ile kolayca uygulama geliştirme imkânı ve donanımsal tasarımcılara işletim sisteminin rahatlıkla kullanabileceği bir arayüz oluşturur.

Gerçek zamanlı çalışan sumo robot üzerinde, gerçek zamanlı işletim sisteminin çalışma yapısına göre, ana işi bölümlere (iş parçalarına) ayırabilmek ve/veya çoklu görev yönetimi oluşturabilmek mümkündür. Bunun için; Inam, Mäki-Turja, ve Sjödin gerçek zaman uygulamalarında FreeRTOS uygulamalarını kullanmışlardır (8). İşletim sistemlerinin temel görevi; uygulama programlarının ihtiyaçlarını karşılamak amacı ile donanım kaynaklarının yönetilmesini

sağlamaktır. Bu amacı, bazı bileşenleri kullanarak gerçekleştirir. Bunlar; çoklu görev (multitasking), senkronizasyon (synchronization), kesme ve olay yönetimi (interrupt and event handling), giriş/çıkış (input/output) işlemleri, iç görev iletişimi (inter task communication), zamanlama ve saat (timers ve clocks) yönetimi ve bellek yönetimi (memory management)' dir. Güncel işletim sistemleri yukarıdaki bileşenlerden pek çoğunu kullanabilmektedir. VxWorks, Integrity, LinuxRT bunlar arasında sayılabilir. İşletim sistemleri "görev" ve "öncelik" tercihlerine ilişkin altyapıları içermekte ve çeşitli görev sıralaması mekanizmalarını sağlamaktadır. Tercih edilen işletim sistemine göre, bu bileşenler çalışma zamanının başında belirlenerek, yazılımın çalıştığı süre boyunca sabit olarak uygulanabilir. Bunun yanı sıra, işletim sisteminin türüne göre dinamik yapılar olarak uygulamalar çalışırken kaynakların dağıtım modelleri de değişim gösterebilmektedir. Yazılımsal olarak iş paylaştırma ve koordineli çalışma algoritmaları Microsoft Windows işletim sistemi türevleri üzerinde çalışır. Özellikle robot kontrolü için kullanılan Microsoft Developer Studio (MRDS) yazılımında, işletim sistemi çekirdeği (kernel) içerisinde yer alan "Concurrency and Coordination Runtime (CCR)" ve "Distributed Software Services (DSS)" alt yapıları bulunmaktadır. Windows ortamında çok işlemcili ve çok çekirdekli donanımsal yapılar kullanıldığında; işletim sistemi çekirdeği, bahsi geçen yazılımsal yapılarla sağlanmakta ve kullanıcı müdahale etmemektedir. Windows platformunda manuel olarak birden çok işlemci ya da çekirdeğe iş paylaşımı yaptırılmak istenirse, geliştirilecek yazılımların bu yapıya uygun çalışması için yazılımcı tarafından bu işlemlerin yapılması gerekmektedir.

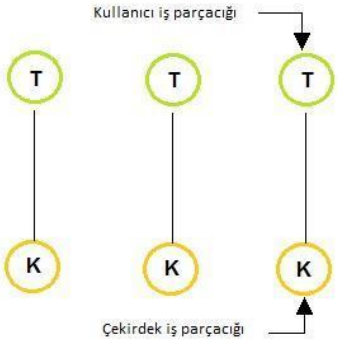
Çalışmada kullanılan gerçek zamanlı çalışan sumo robot üzerinde, Arduino Mega kontrol kartı ile çalışabilen FreeRTOS kütüphanesi tercih edilmiştir. Bu yapıya benzer başka çalışmalarda kullanılan, işletim sistemlerinde çoklu iş parçası (multithreading) çalışma modelleri de bulunmaktadır. Bu modellere göre yapılar; çoktan bire (many to one), bire bir (one to one) ve çoktan çoğa (many to many) olmak üzere üç tiptir. Çoklu iş parçacığı yapısında olan (multithreading) işler; kullanıcıdan gelen talepler doğrultusunda, Şekil 1, Şekil 2 ve Şekil 3' de yer alan üç modelden birine göre ana iş üzerinde yerini alır (9).

Şekil 1' de görülen çoktan bire modelinde, kullanıcı seviyesindeki birçok iş parçacığının tümü işletim sistemi çekirdek seviyesinde tek bir iş parçacığı üzerinde toplanır. İş parçacıkları yönetimi kullanıcı uzayında iş parçası yönetimi kütüphanesi tarafından yapılır.



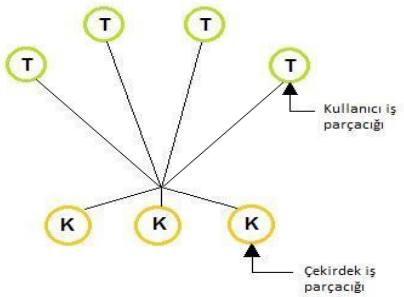
Şekil 1. Çoktan bire (Many to one) iş modeli (9).

Bire bir model, her kullanıcı iş parçacığını işlemek için ayrı bir çekirdek iş parçacığı oluşturur. Bu modelin uygulamalarının çoğu, oluşturulabilecek iş parçacığı sayısına sınır koyar. Linux ve Windows 95' den XP' ye kadar pek çok işletim sistemi, iş parçacıkları için birebir modeli kullanır. Bu model şeması Şekil 2' de gösterilmiştir.



Şekil 2. Bire bir (one to one) iş modeli (9).

Çoktan çoğa model, bire bir ve çoktan bire modellerin en iyi özelliklerini birleştirerek, herhangi bir sayıdaki iş parçasını eşit veya daha az sayıda çekirdek iş parçasına çoğaltır. Kullanıcılar istediği sayıda iş parçacığı oluşturabilir. Çekirdek kaynaklı sistem çağrıları, bu modelde devam eden işleri veya tüm süreci engellemez. İşler birden fazla işlemciye bölünebilir. Bu model şeması Şekil 3'de gösterilmiştir.

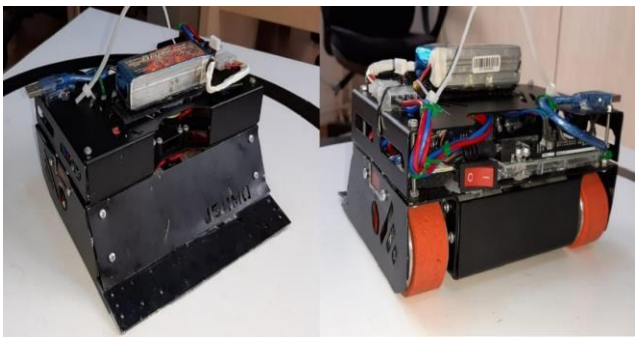


Şekil 3. Çoktan çoğa (many to many) iş modeli (9).

Çoktan çoğa iş parçacığı modelinde, çoklu işlemci veya çoklu çekirdek yapısı kullanıldığında, bilgi işleme performansı önemli ölçüde artarak ve daha verimli kullanılabilecek hale dönüşür. Bu işlem robotik uygulamalarında daha hızlı karar verme yeteneği, daha hızlı tepki süresi ve eş zamanlı pek çok işi yapabilme yeteneği kazandırılabilir.

3. Deneysel Çalışmalar

Çalışma kapsamında geliştirilen gerçek zamanlı işletim sistemi ile çalışan sumo robotun donanımsal yapısı Şekil 4' de görülmektedir.



Şekil 4. Gerçek zamanlı işletim sistemi ile çalışan sumo robot

Şekil 4' de görüldüğü üzere mekanik ve elektronik bileşenler bir araya getirildikten sonra, kontrol kartının USB bağlantı konektörü ile bilgisayardan Arduino programlama arayüzü ile yazılımın yüklemesi yapılmıştır.

Gerçek zamanlı işletim sistemine sahip sumo robot, hem gerçek zamanlı işletim sistemi ile hem de Arduino komut sistemi ile belirli görevleri yapması için programlanmıştır. Bunun için her iki programlamada sonucu somut olarak gözlemleyebileceğimiz işler seçilmiştir. Test için seçilen ve farklı fonksiyonları içeren iş yükü hesaplaması yaptırılmıştır. FreeRTOS kütüphanesi kullanılarak ve kullanılmadan aynı işlemlerin (hesaplamaların) karşılaştırılabilmesi adına Tablo 1' de yer alan fonksiyonlar (görevler) tanımlanarak uygulanmıştır. İşletim sistemi üzerinde iş yükü testinde işlemleri yerine getirme süreleri yazılımsal olarak ölçülmüştür. Bu ölçümler sonucunda Tablo 1' deki değerler elde edilmiştir.

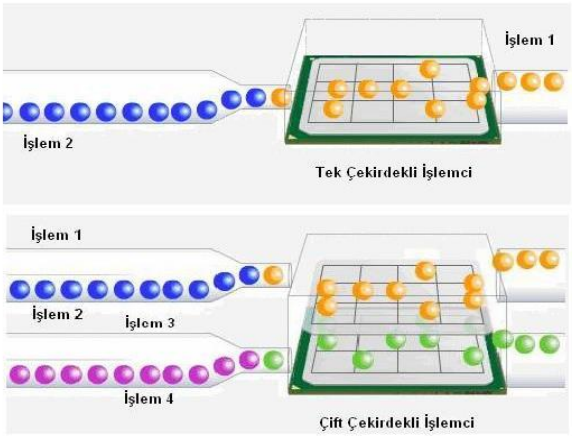
Tablo 1' de birinci sütun çalıştırılan test fonksiyonları, ikinci sütun ise tek görev ile RTOS çalıştırmada ölçülen süreler, üçüncü sütun ise RTOS kütüphanesi kullanılmadan ölçülen süreleri yer almaktadır.

Tablo 1. Görevlerin gerçekleştirme süreleri

Fonksiyon Adı	**RTOS Tek Task (µs)**	**RTOS Olmadan (µs)**
nop	0,063	0,063
DigitalRead	6,355	6,287
DigitalWrite	7,21	7,145
pinMode	4,599	4,53
multiply byte	0,632	0,632
divide byte	5,54	5,535
add byte	0,569	0,569
multiply integer	1,388	1,386
divide integer	14,477	14,457
add integer	0,884	0,883
multiply long	6,36	6,355
divide long	38,912	38,887
add long	1,766	1,765
multiply float	7,747	7,742
divide float	83,537	83,462
add float	9,617	9,607
itoa( )	12,962	12,957
itoa ( )	126,187	125,987
dtostrf [()	81,012	80,887
random ( )	93,187	93,137
y | =(1<<x)	0,569	0,569
bitset ()	0,569	0,569
analogReference ( )	-0,013	-0,013
analogRead ( )	111,987	111,987
analogWrite () PWM	8,817	8,867
delay (1)	1008,487	1007,987
delay (100)	99999,98	100024,98
delayMicroseconds (2)	0,758	0,757
delayMicroseconds (5)	3,779	3,776
delayMicroseconds (100)	99,487	99,337

İşletim sistemi üzerinde iş yükü testinde kullanılanlara benzer fonksiyonlar, gerçekleştirilen kontrol yazılımında; sistem boşta, belirli pinden değer okuma, belirli pine değer yazma vb. görevlerdir. Bu görevler kontrol algoritmasında farklı sensörlerden veri alma, gerekli kontrol sinyallerini üretme ve motorların hareketinde kullanılan sinyallerdir. Kullanılan sistem zaman saatini, hazır fonksiyon (millis) ile bir değişkene aktararak, görevlerin başlangıç ve bitiş zamanları kaydedilmiştir. Bu kayıtlarda yer alan değereler arasındaki farktan yararlanılarak o görevin çalışma süresi elde edilmiştir. Tablo 1' de görüleceği üzere; görevler için gereken süreler birbirine yakın değerlerde çıkmıştır. Oluşturulan yazılımda, gerçek zamanlı işletim sistemi kütüphanesi kullanılmayan yapıda görevler sırayla yapılmakta ve bir görev bitmeden diğerine geçmemektedir.

Gerçek zamanlı işletim sisteminde görev yöneticileri ayrı ayrı olmakla beraber, kullanan Arduino Mega kontrol kartı tek çekirdekli bir işlemciye sahiptir. Bu durum Şekil 5' de ifade edilmiştir.



Şekil 5. Tek çekirdekli işlemci ile çift çekirdekli işlemci yapısı (10).

Bu çalışmada kullanılan robot için, GZİS' lerinde kullanılan algoritma modeli çoklu iş parçası yönetimi (multithreading) modelleri ile de kıyaslanabilir. Benzer şekilde çoklu görev yönetimi (multitasking) yapısının yazılımlarda ve işletim sistemlerinde kullanılmasının başka bir amacı da vardır. Bu, algoritmalarda öncelik ataması yapabilmesine rağmen, çalışmaya hazır olan çok sayıda görev arasında hızlı geçişin sağlanması ve tüm görevlerin en kısa zamanda bitirilmelidir. Bunun için işlemci çekirdeğinin optimum paylaşım süresinde kullanmaktadır (11).

Çalışmada oluşturulan yazılım; Ardiuno Mega kontrol kartının donanımsal kısıtları da göz önüne alındığında, iş paylaştırma yapısı "çoktan bire" modelinin çalışma yapısına benzerlik gösterebilir. Ancak uygulamada yapılan çoklu iş parçacığı modeli değil, çoklu görev yönetimdir. Bu yapıda sensörlerden gelen her bir veri, kullanıcıya ait iş parçaları (user thread) yapısına benzerlik göstermektedir. Kontrol kartının donanımsal kısıtları sebebi ile yazılımın çalışması, çoklu iş parçacığı yapısına benzemekle birlikte öncelikli görev yönetimi yapısı kullanılmıştır. Kullanılan komutlara ait test listesi ve matematiksel fonksiyon yapıları Tablo 1 ve altında yer alan paragrafta açıklanmıştır.

4. Sonuçlar

Görevlerin öncelik sırasının belirlenmesi, kullanılan kontrol kartının işlemcisinde yer alan çekirdek sayısının tek olması, veri yolu transfer hızının düşük olması, iş parçacığı desteğinin olmaması, çalışma frekansının düşük olması vb. donanımsal ve yazılımsal kısıtları sebebi ile beklendiği gibi gerçekleşmemiştir. Görevler belirli öncelik sırasına göre gerçekleştirilse bile, işlerin tek çekirdekli yapıda tek bir kanal üzerinden gerçekleşmesi işlemlerin daha hızlı olmasını sağlamamıştır. Başka bir ifade ile çoklu görev yönetimi modeli kullanıldığında beklenen performans artışı sağlanmamıştır.

Araştırma bulgularından da görüleceği üzere; gerçek zamanlı işletim sistemi ile çalışan sumo robotun, gerçek zamanlı çalışmayan sumo robota göre yüksek performanslı çalışması farklı bileşenlere de bağlıdır. Bunlar; uygun çoklu iş parçacığı modeli, öncelikli görev yönetimi ile yazılımın hazırlanması (seçilecek GZİS' nin desteği olması şartı ile) ve kullanılan kontrol kartının teknik sınırlılıklarıdır. Gerçek zamanlı işletim sistemine sahip sumo robota verilen görevlerin öncelik sırası ve gerçekleştirme süreleri yazılımsal olarak ayarlanabildiği için, gerçek zamanlı işletim sistemine sahip olmayan sumo robota göre yazılımsal anlamda daha esnek özelliklere sahip olduğu görülmüştür. Bu durum daha ayrıntılı açıklanırsa; gerçek zamanlı işletim sistemi ile istenilen iş istenilen zamanda, istenilen süre ile çalıştırabilir. İşlere öncelik atanarak, önce ya da sonra çalışması gereken iş planlanabilir. Çoklu iş parçacığı yönetiminde, bir işi birden fazla alt işe bölüp, bu bölümleri birbirinden bağımsız olarak çalıştırabilme sağlanabilir.

Öncelikli çoklu görev yönetimi, bu çalışmada donanımda yer alan 5+2 sensörden alınan giriş verileri üzerinde yazılımsal önceliklendirilerek uygulanmıştır. Bu yedi ayrı sensör giriş verisi farklı şekilde önceliklendirildiğinde yazılımsal problemler yaşanmıştır. Bunun sebebi; kontrol kartının donanımsal ve yazılımsal sınırlılıkları sebebi ile veri işleme hızına cevap vermemesidir. Bu da önceliklendirme işleminin oluşturduğu dinamik iş yüküne cevap vermemesi anlamına gelir. Bu durum; robotun yerine getirmesi gereken görevleri karıştırarak, sürekli değişen dinamik duruma cevap vermeyerek, kararsız olmasına neden olmuştur. Bu sorun görev önceliklendirilmesinin dört sensör verisinin öncelikli olarak işlenmesi ile (öncelik sayısının dörde düşürülmesi ile) çözülmüştür. Bu çözüm sonrası robotun yapması beklenen doğru hareketleri yerine getirdiği görülmüştür.

Bu çalışmada robotun hızlı algılaması ve karar vermesi için, çoktan bire çoklu iş parçacığı yapısına benzetilebilecek öncelikli görev yönetimi kullanılmıştır. Robot; bir işlemcili ve tek çekirdekli sistem üzerinde, kontrast sensörleri ve cisim algılamada kullanılan mesafe sensörlerinin önceliğini belirlemiştir. Böylece dohyo sınırlarının algılaması ya da rakibinin algılaması arasındaki önceliği belirlemesi sağlanmıştır. Aynı zamanda da bu görevlerin ne kadar süre ile çalışmaları gerektiği belirlemiştir. Farklı uygulamalarda seçilecek kontrol kartında kullanılacak işlemcinin çalışma

frekansı yükseltilmesi de birim zamanda yapılacak iş miktarını arttıracaktır. Böylelikle daha hızlı karar verme ve harekete geçme gibi veri işlemede performans artışı sağlayabilecektir. Fakat farklı kontrol kartları veya işlemci seçiminde yazılımsal uyumluluklar, donanımsal sınırlılıklar ve maliyetler de göz önünde bulundurulmalıdır.

Farklı yoğun hesaplama ve iş yükü gerektirecek daha gelişmiş robotik uygulamalarında, birden çoklu işlemci yönetimi (multiprocessing) ya da çoklu çekirdek üzerinde uygun iş parçacığı modeli (multithreading) birlikte de tercih edilebilir. Benzer uygulamalar farklı yazılımlar kullanılarak, sunucular ve iş istasyonlarında yük dengeleme gibi yazılımsal destekler de eklenerek, farklı işletim sistemleri ile kullanılabilir. Bu durum pek çok endüstriyel otomasyon ve robotik uygulamasında görülmektedir. Bunun yanı sıra koordinasyon ve dağıtık yazılım servisleri de kullanıldığında yazılımsal yeni esneklikler sağlanabilir. Ayrıca donanımsal bileşenler ele alındığında; seçilecek kontrol kartının, sensörlerin tepki süresi, motorların devir sayısı ve gücü, motor sürücülerinin özellikleri ve desteklediği yazılımsal teknolojiler, anlık yüksek akım verebilecek bataryalar seçilerek farklı sumo robot uygulamaları gerçekleştirilebilir. Fakat bu durumun donanımsal maliyetleri de arttıracağı göz önünde bulundurulmalıdır. İleride yapılacak çalışmalarda farklı donanımlar üzerinde farklı yazılımlar geliştirilerek uygulama geliştirilebilir.

Teşekkür

Bu çalışma; 3616-YL1-13 No' lu Proje ile yüksek lisans tez çalışması olarak, Süleyman Demirel Üniversitesi Bilimsel Araştırma Projeleri Yönetim Birimi Başkanlığı tarafından desteklenmiştir.

Kaynaklar

[1] Katipoğlu G. *Üç serbestlik dereceli robot kolunun pozisyon kontrolü*. Doctoral Dissertation, DEÜ Fen Bilimleri Enstitüsü, İzmir, Turkey, 2013.

[2] Naughton JD. *Futurology and robots: Karel Čapek's RUR., culture, theory and critique*. 28(1), 1984.

[3] Boston Dynamics. https://www.bostondynamics.com/atlas (Erişim Tarihi: 20.05.2019).

[4] Albayrak M, Yüksel Y. 8051 Mikrodenetleyicili bir sumo robot tasarımı ve uygulaması. *Süleyman Demirel Üniversitesi Fen Bilimleri Enstitüsü Dergisi*, 11(1), s.96-102, 2007.

[5] Wei H, Huang Z, Yu Q, Liu M, Guan Y, Tan J. RGMP-ROS: A real-time ROS architecture of hybrid RTOS and GPOS on multi-core processor. *In 2014 IEEE International Conference on Robotics and Automation (ICRA)*, (pp. 2482-2487), 2014.

[6] Yu Q, Wei H, Liu M, Wang T. A Novel multi-OS architecture for robot application. *In 2011 IEEE International Conference on Robotics and Biomimetics,* (pp. 2301-2306), 2011.

[7] Robotistan.com. https://www.robotistan.com/orjinal-arduino-mega-2560-r3-yeni-versiyon-1 (Erişim tarihi: 13.04.2018).

[8] Inam R, Mäki-Turja J, Sjödin M, Behnam M. Hard real-time support for hierarchical scheduling in FreeRTOS, *In 23rd Euromicro Conference on Real-Time Systems*, pp. 51-60, 2011.

[9] Techtud, Multithread Modelleri. https://www.techtud.com/short-notes/multithreading-models-many-one-model one-one-model-many-many-model (Erişim tarihi: 20.04.2019).

[10] İTÜ Bilgi işlem Daire Başkanlığı. http://bidb.itu.edu.tr/seyir-defteri/blog/2013/09/06/%C3%A7ift-%C3%A7ekirdekli-i%C5%9Flemci (Erişim Tarihi: 05.10.2018).

[11] Agypt B, Rubin BA, Spivack AJ. Thinking outside the clocks: The effect of layered-task time on the creative climate of meetings. *The Journal of Creative Behavior*, 46(2), pp.77-98, 2012.