ft robotik
fischertechnik

ROBOTİK LABORATUVARI

LİSE

Programlama Dilleri:

C, C++

python

OPEN ROBERTA

ETKİNLİK KİTABI 2

Mobil Robotlar

STEM

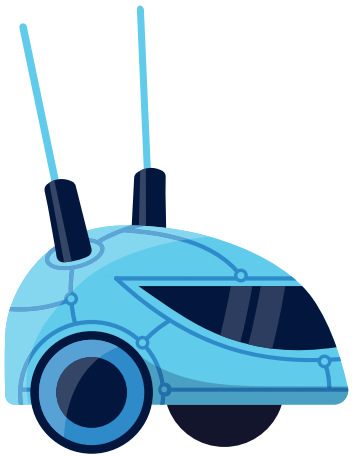
MATHEMATIK
INFORMATIK
NATURWISSENSCHAFT
TECHNIK

İçindekiler

GİRİŞ:

Mobil Robotlar

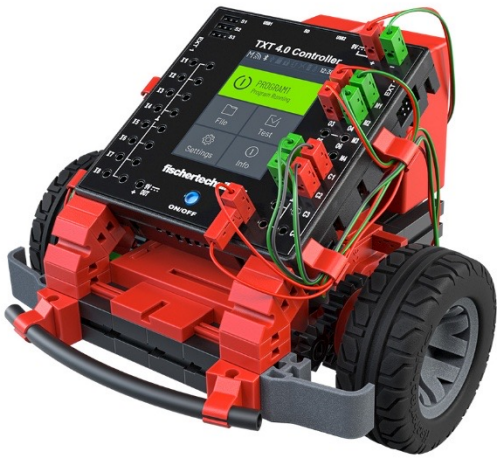
Sayfa: 6



Bölüm 1:

Buggy Mobil Robot

Sayfa: 9



Bölüm 2:

Engel Algılayıcı Buggy

Sayfa: 27

İçindekiler

Bölüm 3:

Hedef Bulucu Buggy

Sayfa: **37**

Bölüm 4:

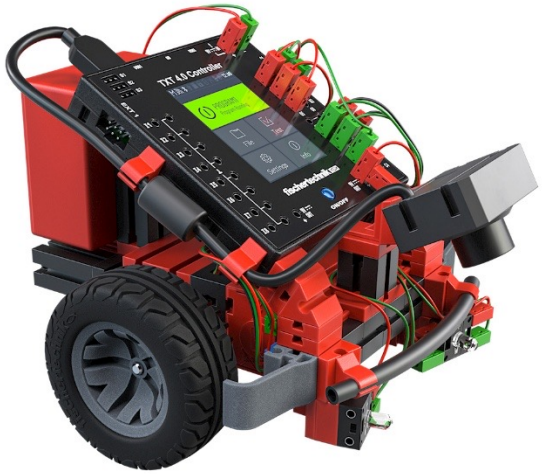
Çizim Robotu

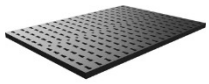
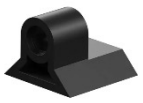
Sayfa: **49**

Bölüm 5:

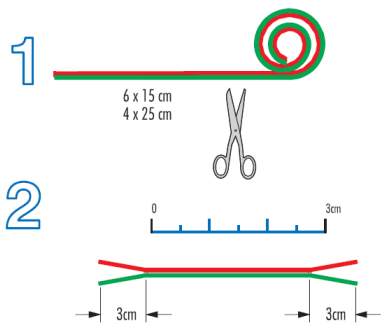
İz Sürücü

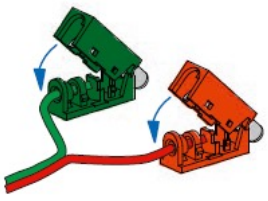
Sayfa: **70**



	163 206 2x		163 205 2x		32 880 1x		153 422 2x
	31 982 10x		35 033 2x		32 985 1x		137 125 2x
	185 360 2x		36 573 1x		36 134 1x		152 522 1x
	31 690 1x		36 950 4x		37 783 2x		172 804 1x
	37 468 7x		78 728 2x		127 421 1x		172 805 1x
	185 789 1x		35 608 1x		31 360 1x		36 437 1x
	133 009 1x		128 598 1x		181 583 21x		173 067 1x
	35 537 1x		522 460 1x		181 584 28x		134 867 1x

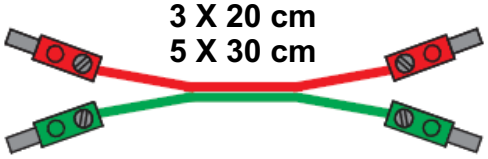
Kablo Bağlantıları





Hazır kablolar

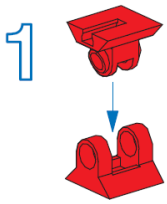
3 X 20 cm
5 X 30 cm



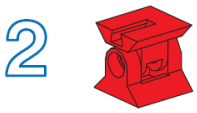
1 X 30 cm

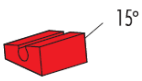
Kurulum Hazırlık

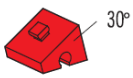
2x Teker

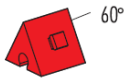


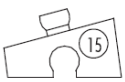
2x

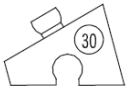


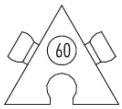












Açılı Bloklar

1x

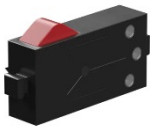
INPUT - Sensörler

Ultrasonik Mesafe Sensörü

IR İz Sensörü

USB Kamera

NTC Isı Sensörü



Touch (anantar) Sensör

Foto-transistör Sensör

OUTPUT – Motor / Lamba

Enkoder Motor

Lamba

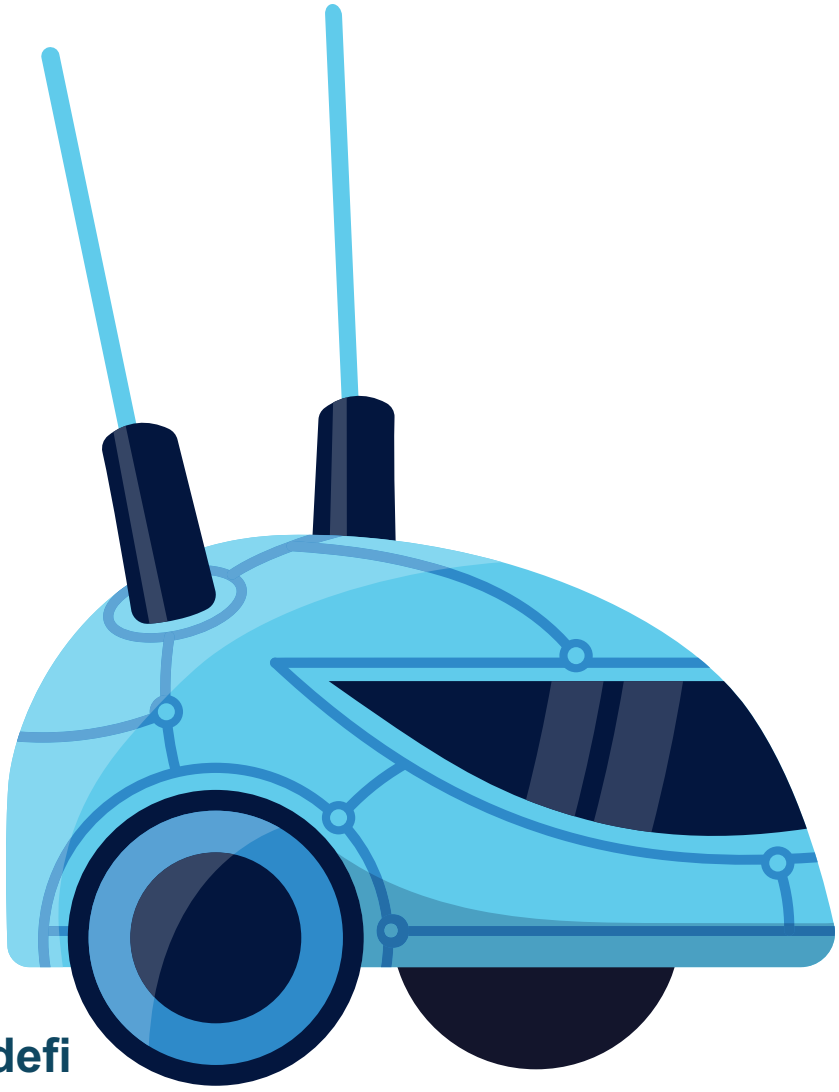
Lens Uçlu Lamba

ftrobotik

fischertechnik

GİRİŞ

Mobil Robotlar



Öğrenme hedefi

- ✓ Mobil Robot Nedir?
- ✓ Mobil Robot Örnekleri
- ✓ Kitapta Neler Öğreneceğim?

Mobil Robotlar

Mobil Robot Nedir?

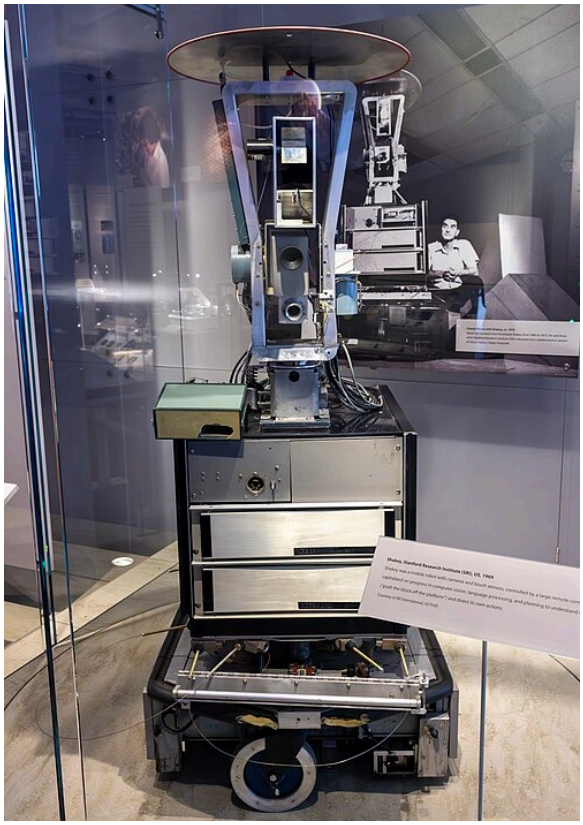
Mobil robotlar, bulunduğu ortamda kendi başına hareket edebilen ve çeşitli görevleri yerine getirebilen akıllı sistemlerdir. Bu sistemler, motorlar yardımıyla hareket eder; sensörler aracılığıyla çevrelerini algılar; ve mikrodenetleyiciler ile karar vererek etkileşimde bulunurlar. Kimi zaman sadece bir çizgiyi takip eden basit yapılar olabilirken, kimi zaman karmaşık algoritmalarla otonom çalışan araçlara dönüşebilirler.

Örnekler:

- Çizgi izleyen robot
- Otonom araçlar
- Depo içi taşıma robotları
- Keşif robotları

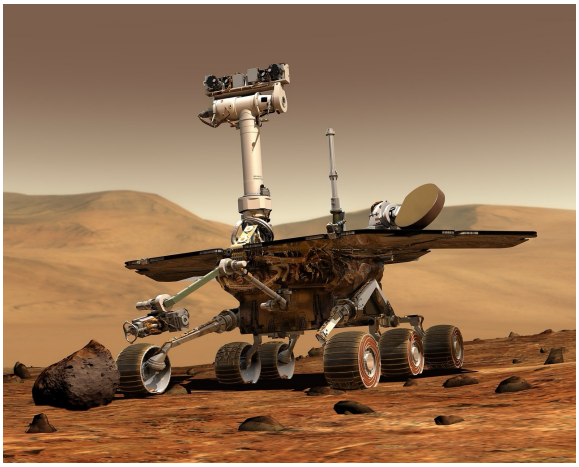
Mobil robotların geçmişi, 20. yüzyılın ortalarına dayanır. 1960'larda NASA için geliştirilen **Shakey the Robot**, ilk otonom mobil robotlardan biri olarak kabul edilir. Shakey, ortamını haritalandırabilen, plan yapabilen ve karar verebilen bir yapıdaydı.

Günümüz dünyasında, mobil robotlar; endüstriyel üretimden otonom araçlara, akıllı tarım sistemlerinden sağlık hizmetlerine kadar pek çok alanda aktif olarak kullanılmakta ve teknolojik gelişimin en görünür örneklerinden biri haline gelmektedir. Hareket kabiliyeti, çevresel algılaması ve karar verme yetisi ile mobil robotlar, sadece teknik bilgi değil aynı zamanda analitik düşünme, problem çözme ve sistematik yaklaşım becerileri de gerektiren çok yönlü bir öğrenme alanı sunar.



"SRI Shakey robot, 1969, Computer History Museum" görseli, Bilgisayar Tarihi Müzesi tarafından sağlanmış olup Wikimedia Commons üzerinden paylaşılmıştır. Lisans: Creative Commons Attribution 2.0 Generic (CC BY 2.0). https://commons.wikimedia.org/wiki/File:SRI_Shakey_robot,_1969,_Computer_History_Museum.jpg

fischertechnik AGV



Mars rover

Bölümlerin Tanıtımı

Bu Kitapta Neler Öğreneceğim?

Bu kitap, lise düzeyindeki öğrencilerin mobil robotların temel yapılarını, hareket prensiplerini ve sensör temelli etkileşim yeteneklerini deneyimleyerek öğrenmeleri amacıyla hazırlanmıştır. Her bölümde adım adım ilerleyen yapılar, öğrencilere gerçek dünya problemlerine çözüm üretebilen robotlar geliştirme fırsatı sunar.

1. bölümde, öğrenciler iki enkoder motorlu **Buggy Mobil Robot** ile eşzamanlı motor kontrolünü öğrenirken, hareketin hassas şekilde yönetilmesinin temelini kavrayacaklardır.

2. bölümde, bir adım öteye geçerek sensör temelli etkileşimi tanıyacak ve **Engel Algılayıcı Buggy** ile çizgisel algılama (yol çizgilerini iz sensörü ile algılayarak yolda kalacak) ve fiziksel temasla engel tespiti üzerinde çalışacaklardır.

3. bölümde, **Hedef Bulucu Buggy** sayesinde, temassız engel algılama için **ultrasonik sensör**lerin nasıl kullanıldığını uygulamalı olarak keşfedeceklerdir.

4. bölümde, **Çizim Robotu** projesi ile hassas motor kontrolünü geometrik şekillerin çizimi üzerinden deneyimleyerek, algoritmik düşünme ile görsel çıktılar üretmeyi birleştireceklerdir.

5. bölümde, klasik bir mobil robot uygulaması olan **Dijital İz Sürücü Robot** ile öğrenciler, iz takibi konusunda temel kazanımlar elde edeceklerdir.

6. ve son bölümde ise, bu temel bilgiler **Mesafe Sensörlü İz Takip Robotu** ile birleştirilecek; hem fiziksel engellerin ultrasonik sensörle algılanması hem de kamera ile renk değerlendirmesi gibi ileri düzey uygulamalara giriş yapılacaktır.

Bu Kitaptaki Mobil Robot Türleri

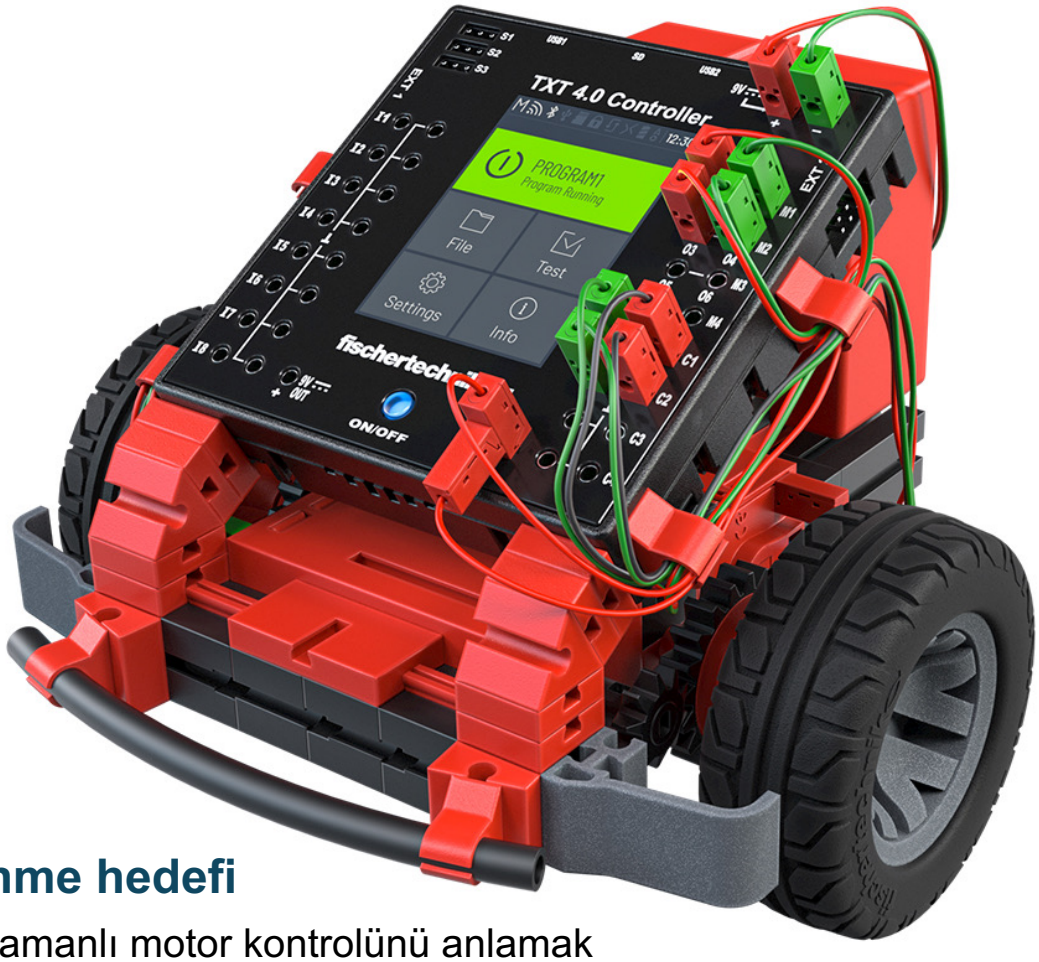
Bölüm	Robot Adı	Temel Kavram
1	Buggy Mobil Robot	Eşzamanlı motor kontrolü
2	Engel Algılayıcı Buggy	İz sensörü, buton ile engel algılama
3	Hedef Bulucu Buggy	Ultrasonik sensörle mesafe ölçümü
4	Çizim Robotu	Hassas yön kontrolü, şekil çizimi
5	Dijital İz Sürücü Robot	İz sensörü ile dairesel çizgi izleme
6	Mesafe Sensörlü İz Takip Robotu	Kamera ile RGB renk değerlendirmesi, engel algılama

ftrobotik

fischertechnik

BÖLÜM 01

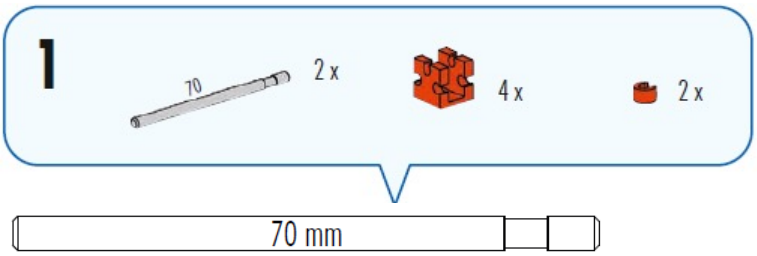
Buggy Mobil Robot

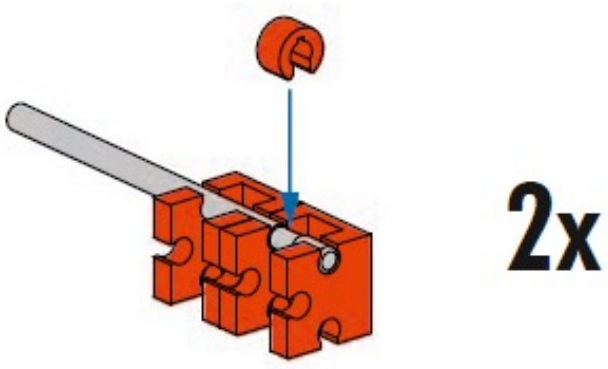


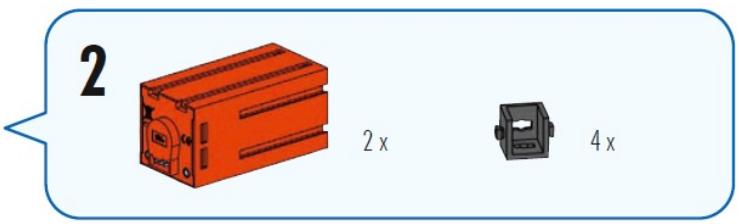
Öğrenme hedefi

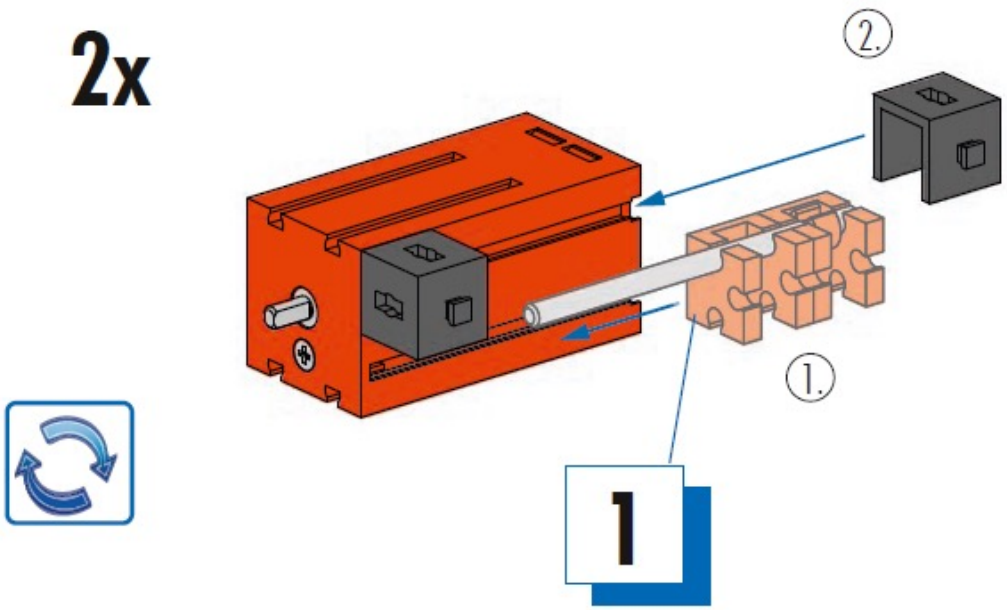
- ✓ Eşzamanlı motor kontrolünü anlamak
- ✓ Sensörler kullanılarak paralel süreçlerin senkronizasyonu
- ✓ Kodlayıcı darbeler aracılığıyla gezinme

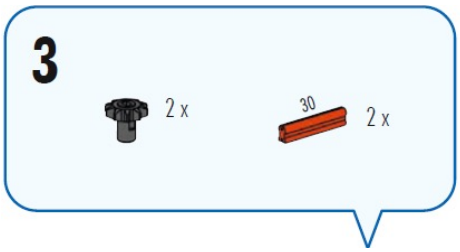
MODEL 8 Buggy Mobil Robot



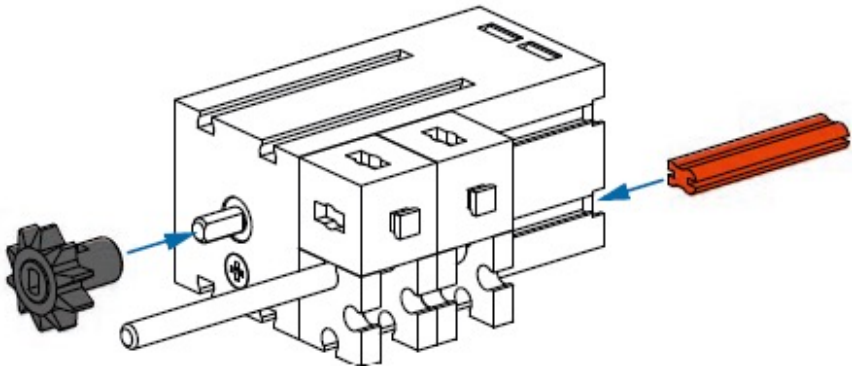


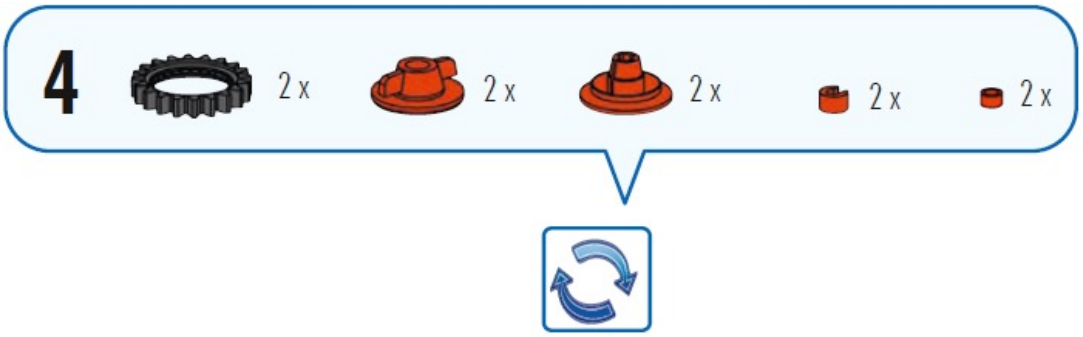




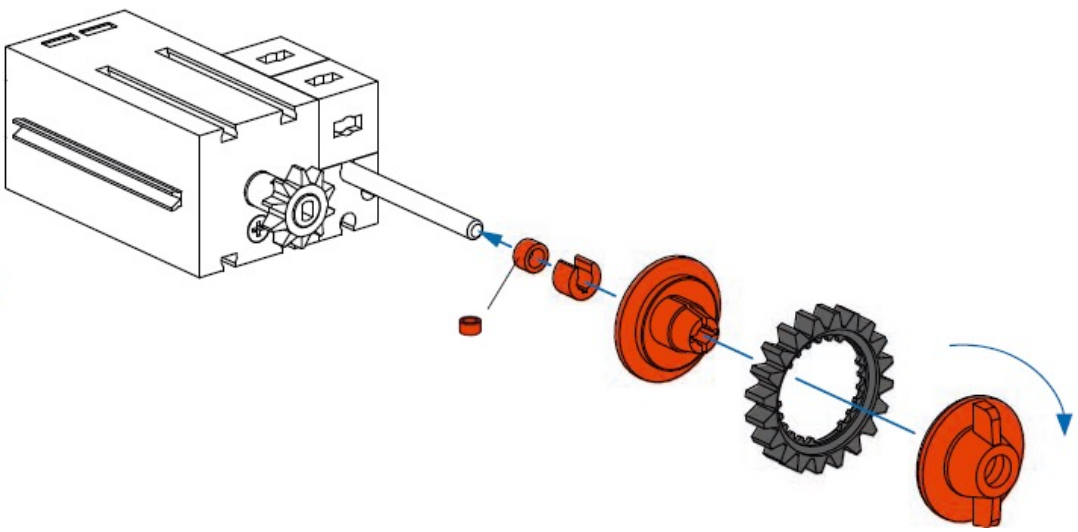


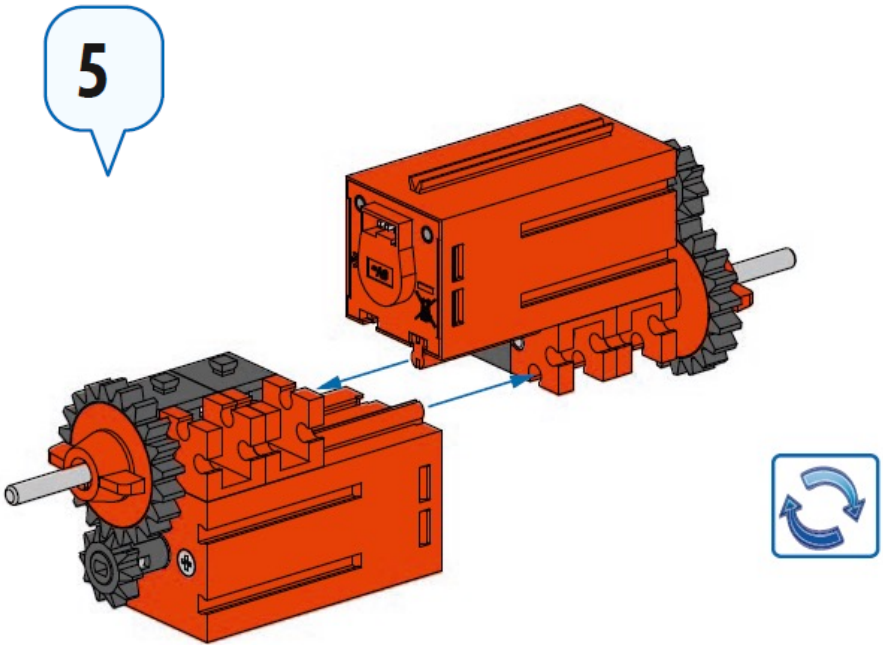
2x

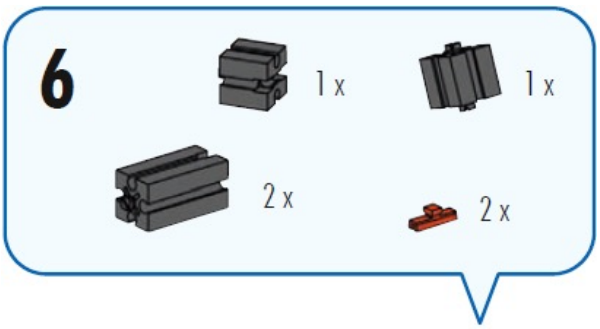


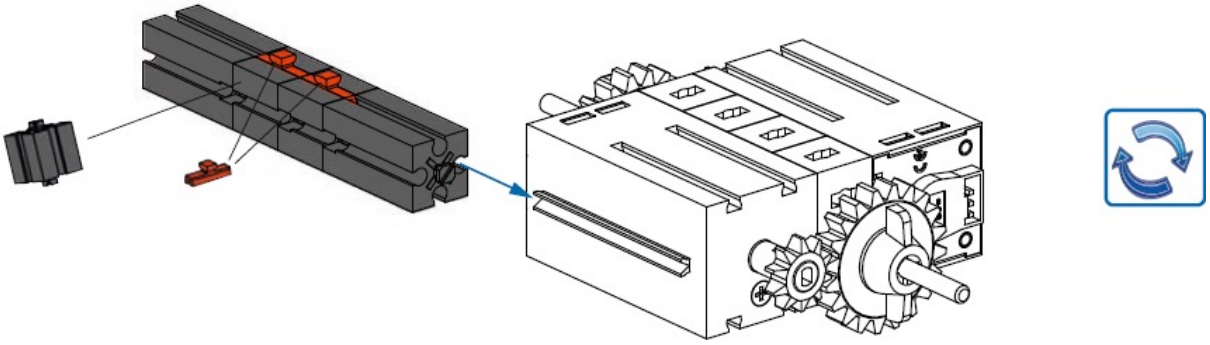


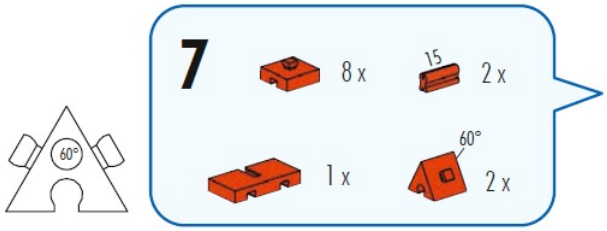
2x

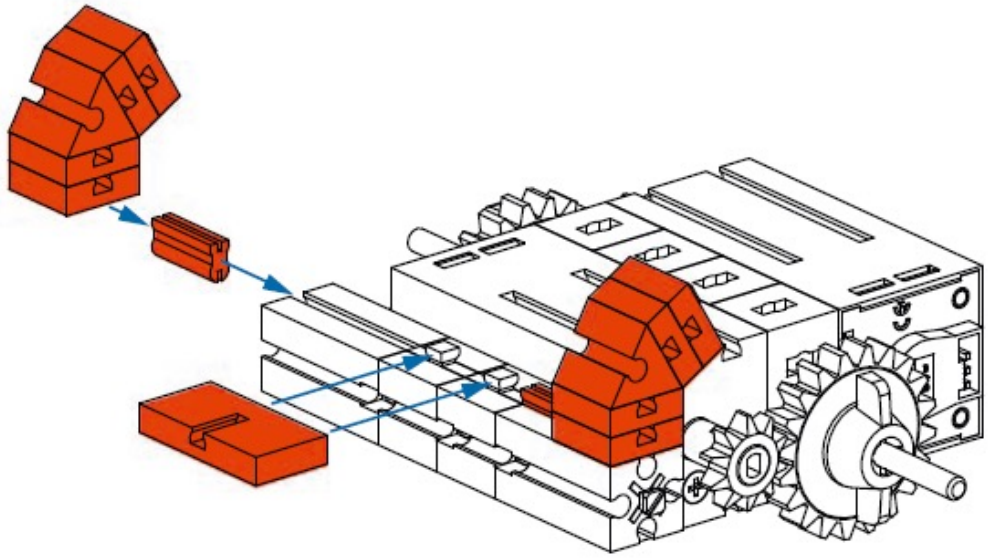


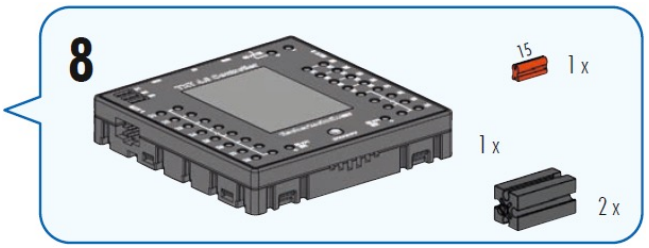

5

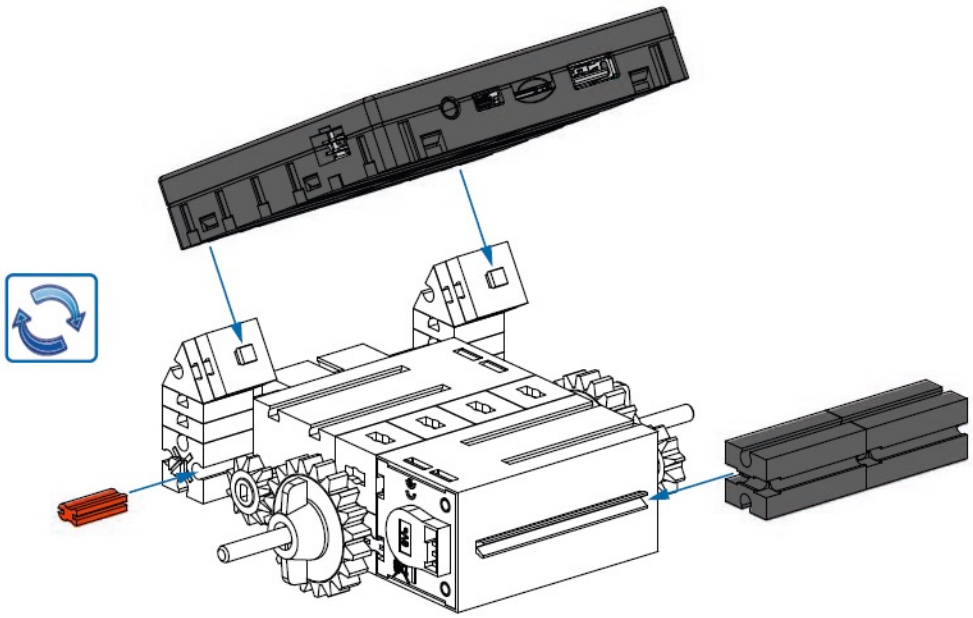

6
1 x
1 x
2 x
2 x

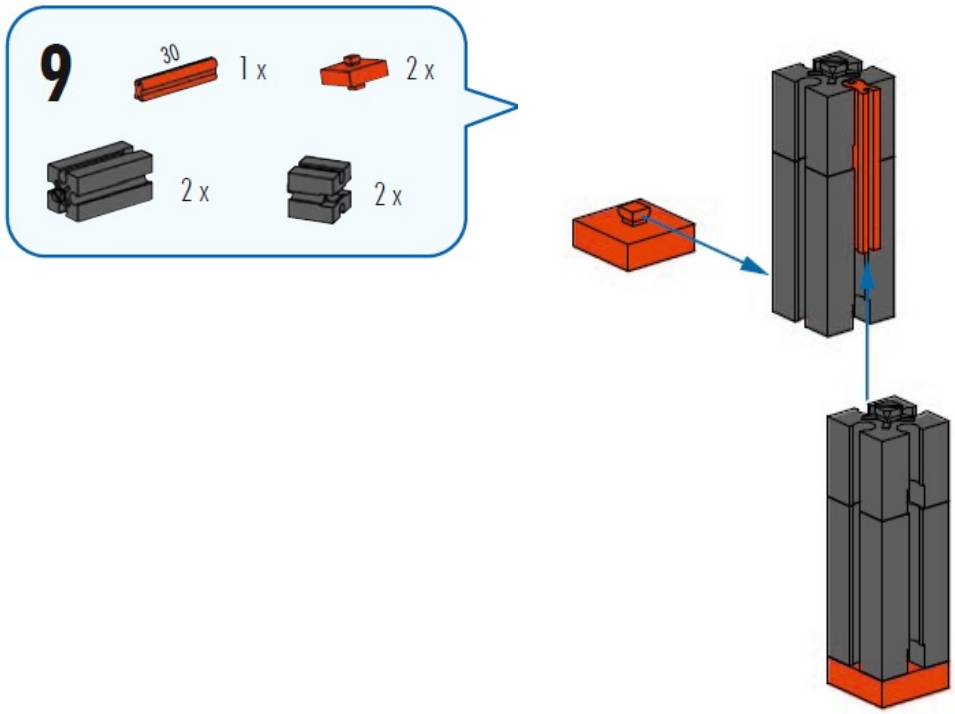



7
8 x
15
2 x
60°
1 x
60°
2 x

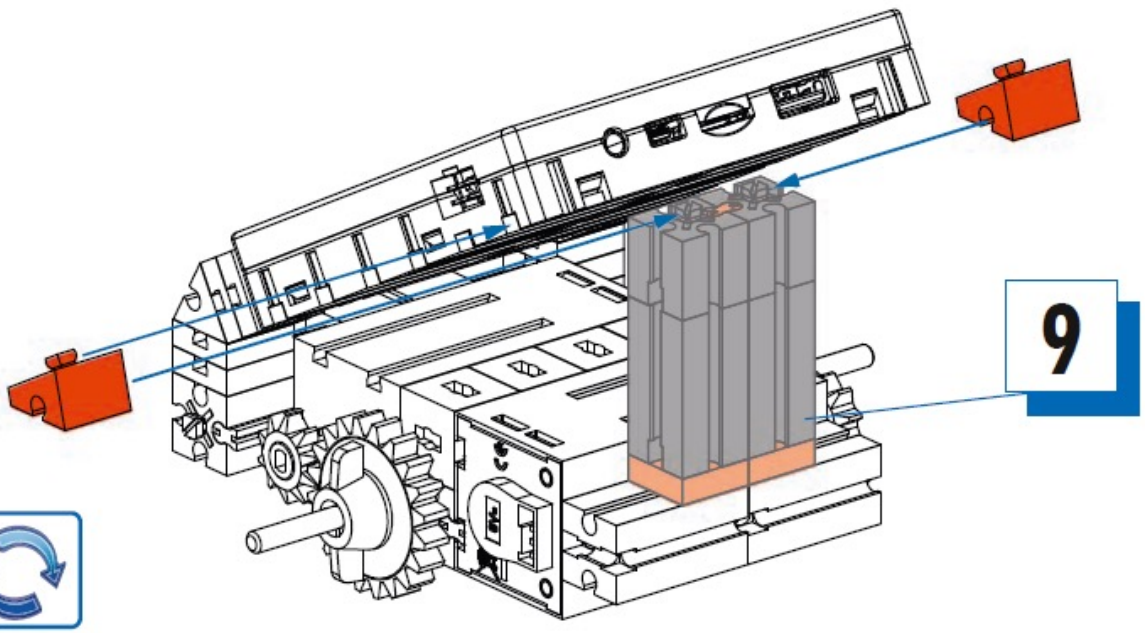


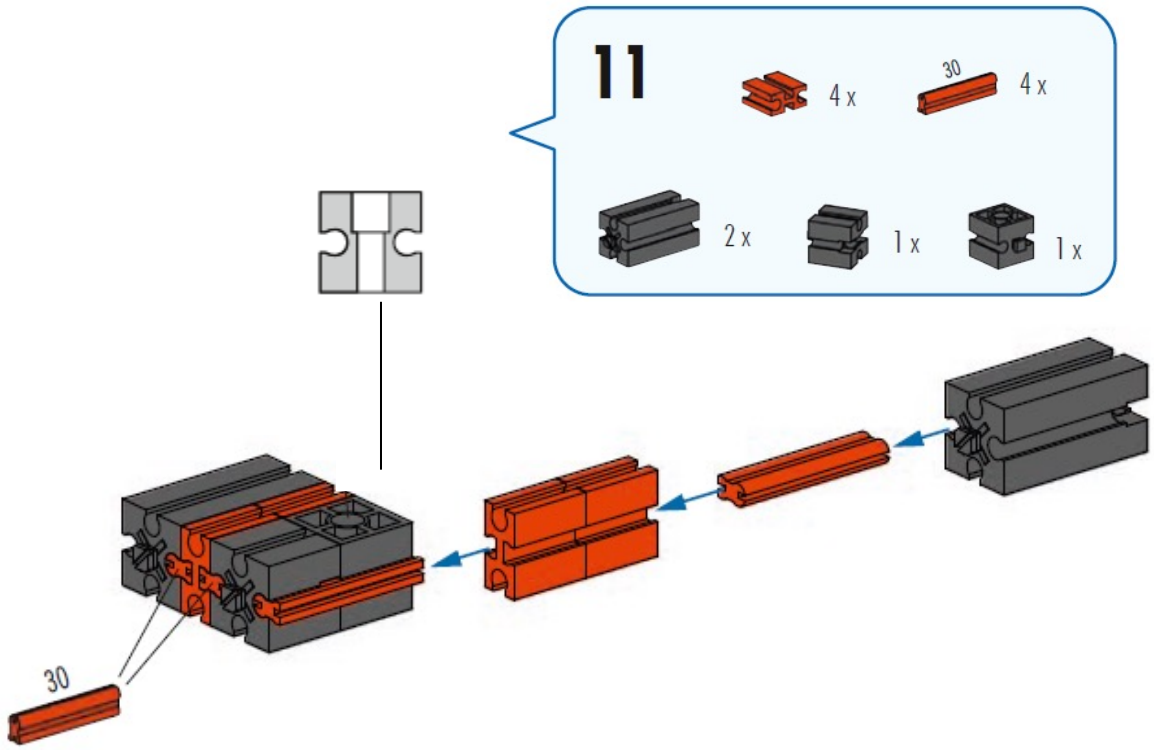

8
15
1 x
1 x
2 x

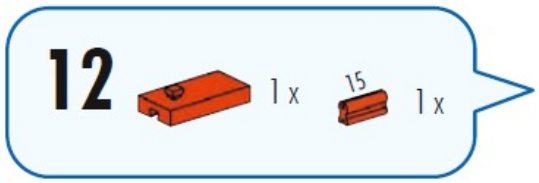


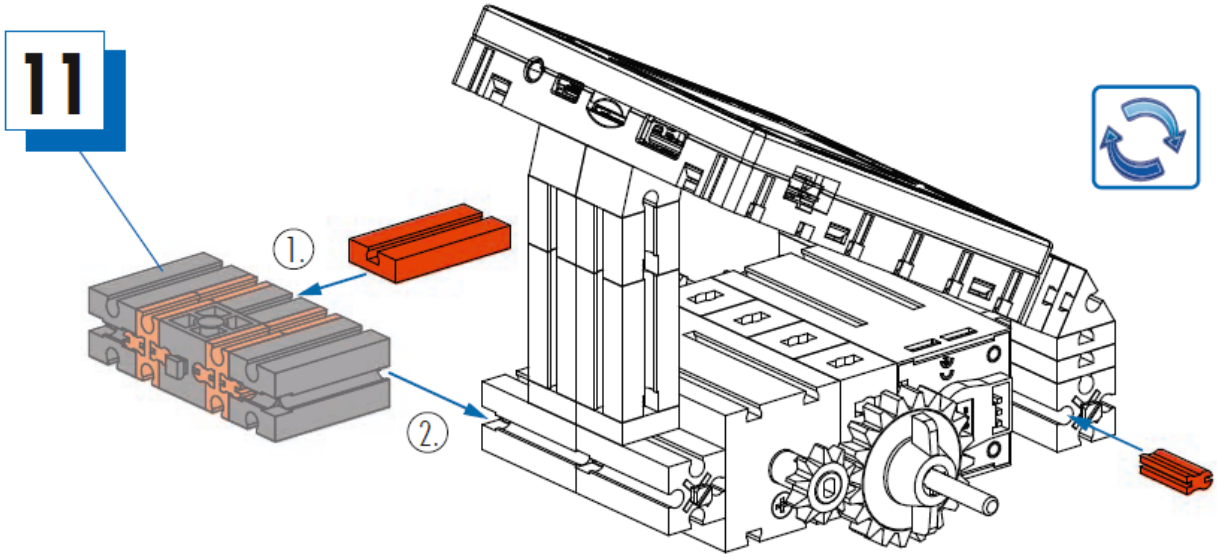

9
30
1 x
2 x
2 x
2 x

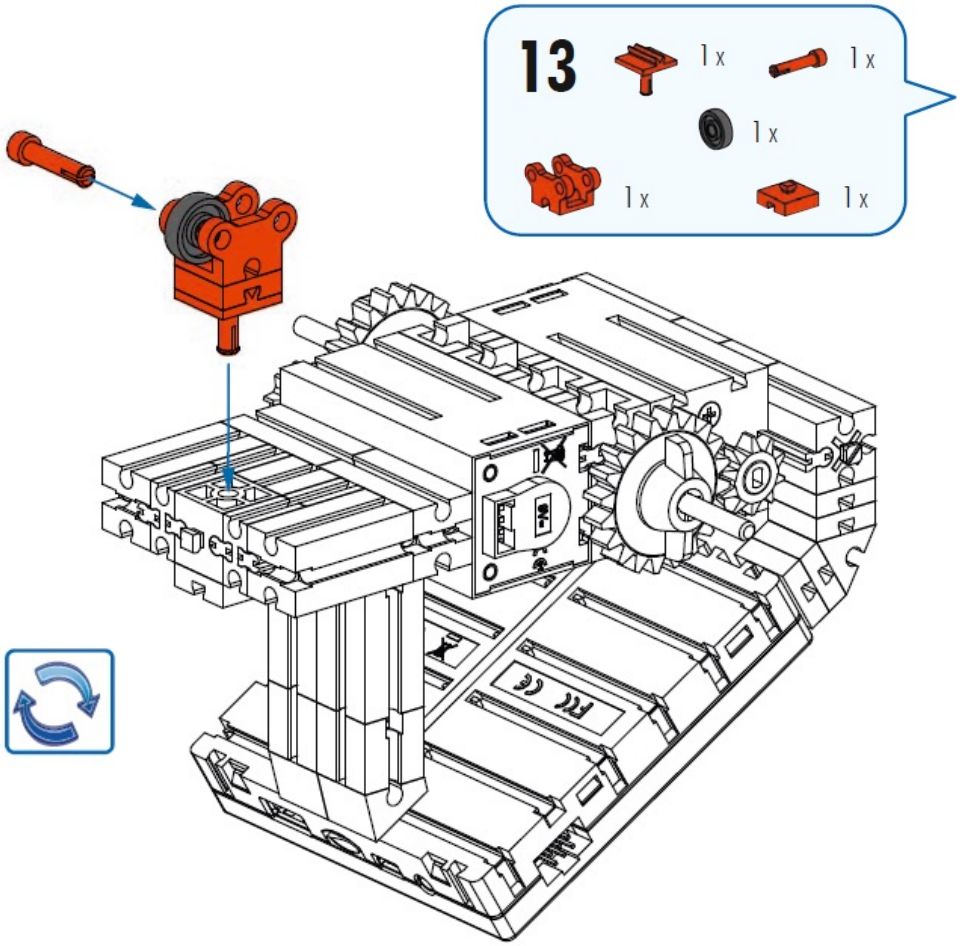

30°
10
30°
2 x

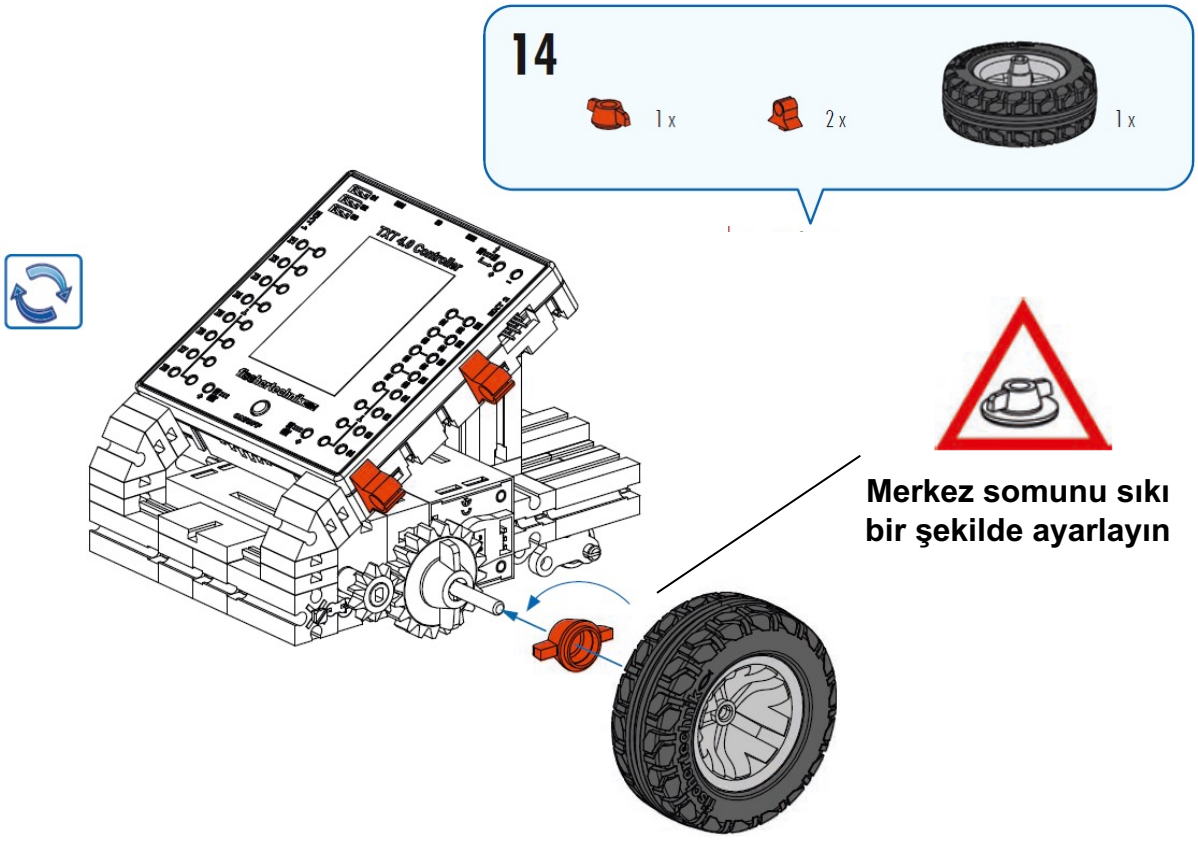

9

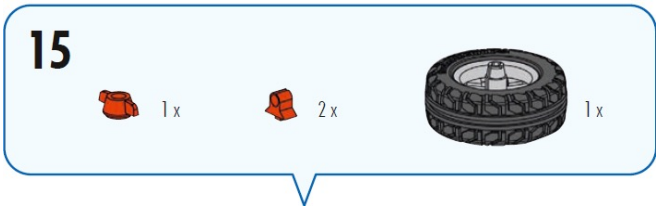

11
4 x
30
4 x
2 x
1 x
1 x
30


12
1 x
15
1 x

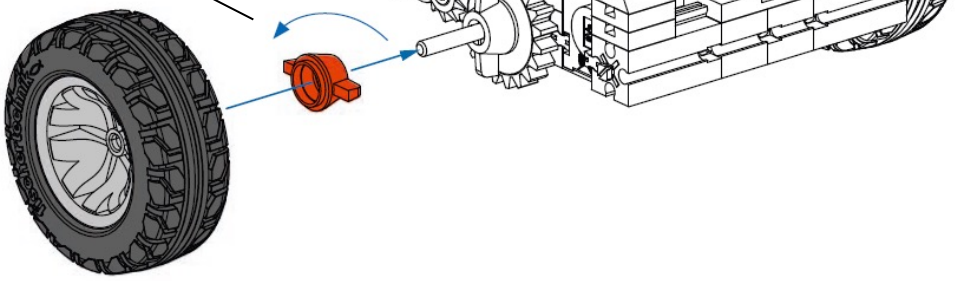

11
1.
2.

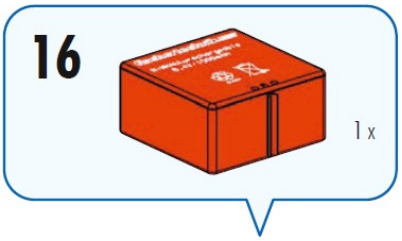

13
1 x
1 x
1 x
1 x
1 x

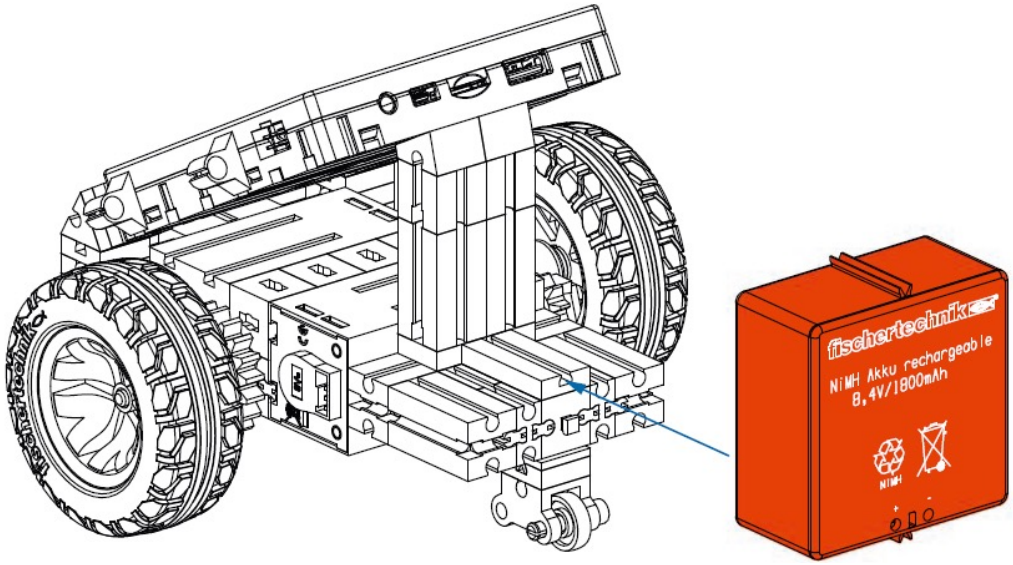

14
1 x
2 x
1 x
Merkez somunu sıkı bir şekilde ayarlayın



Merkez somunu sıkı bir şekilde ayarlayın



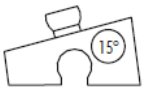


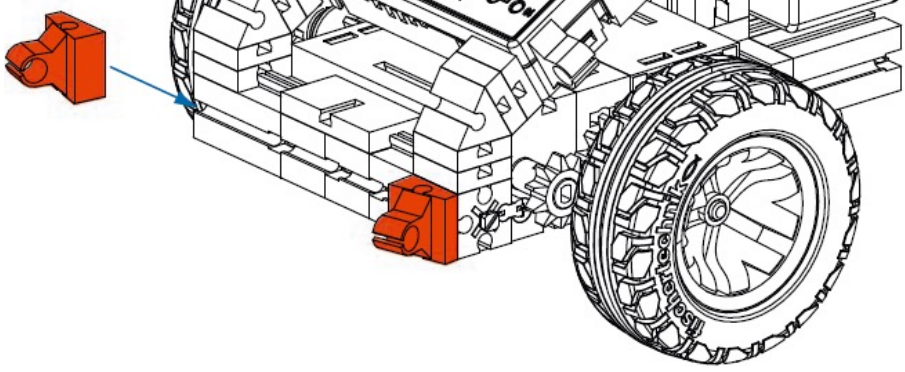


17

2 x


2 x

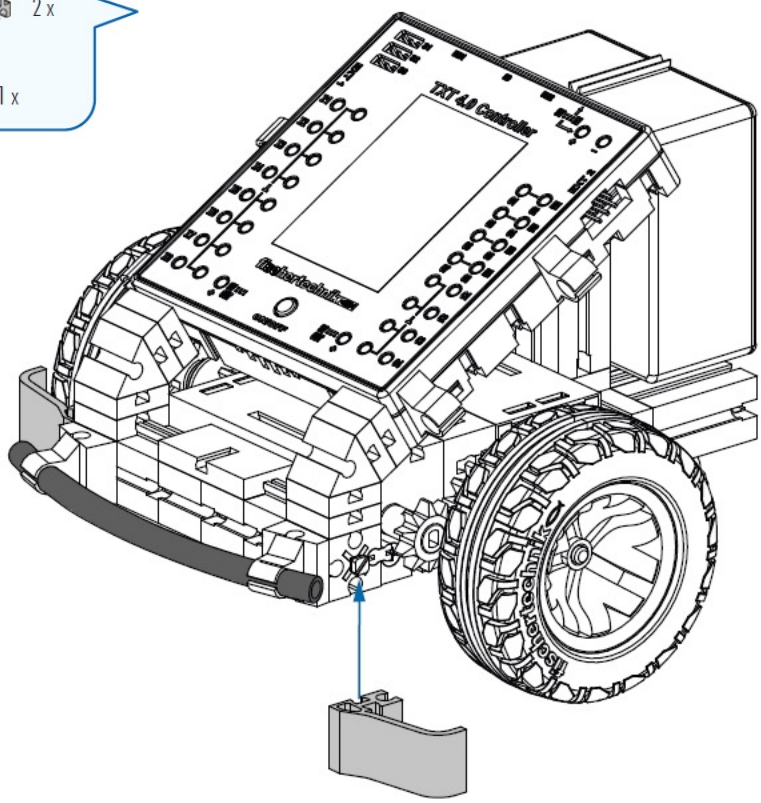




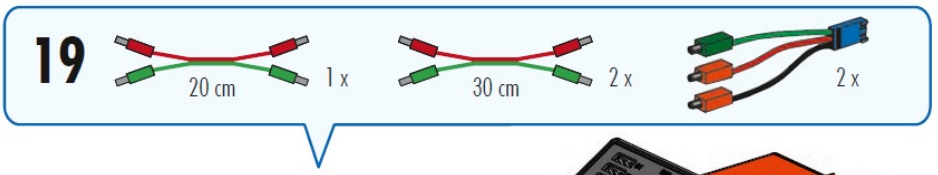
18

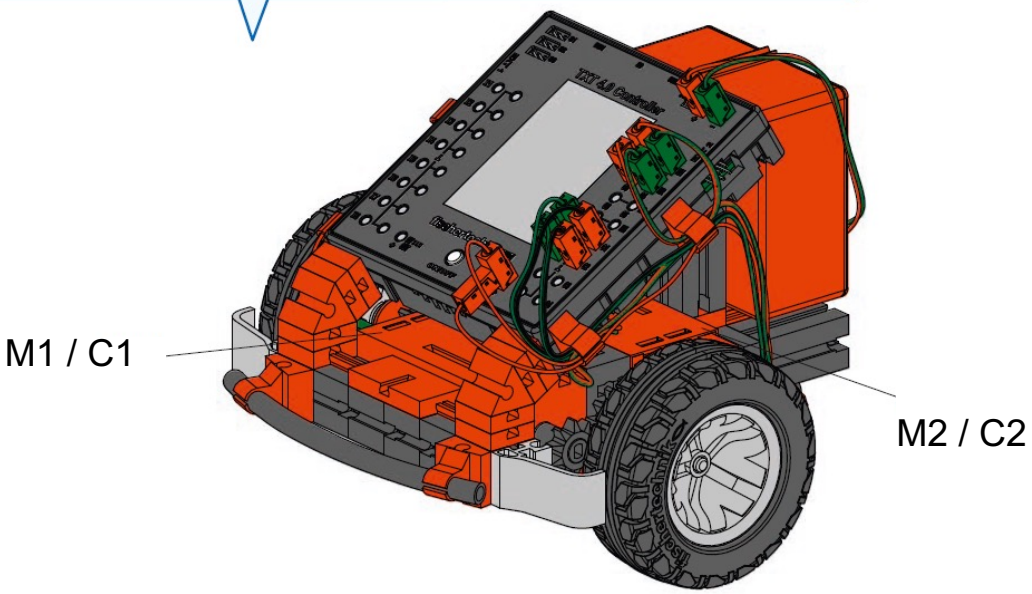

2 x


1 x



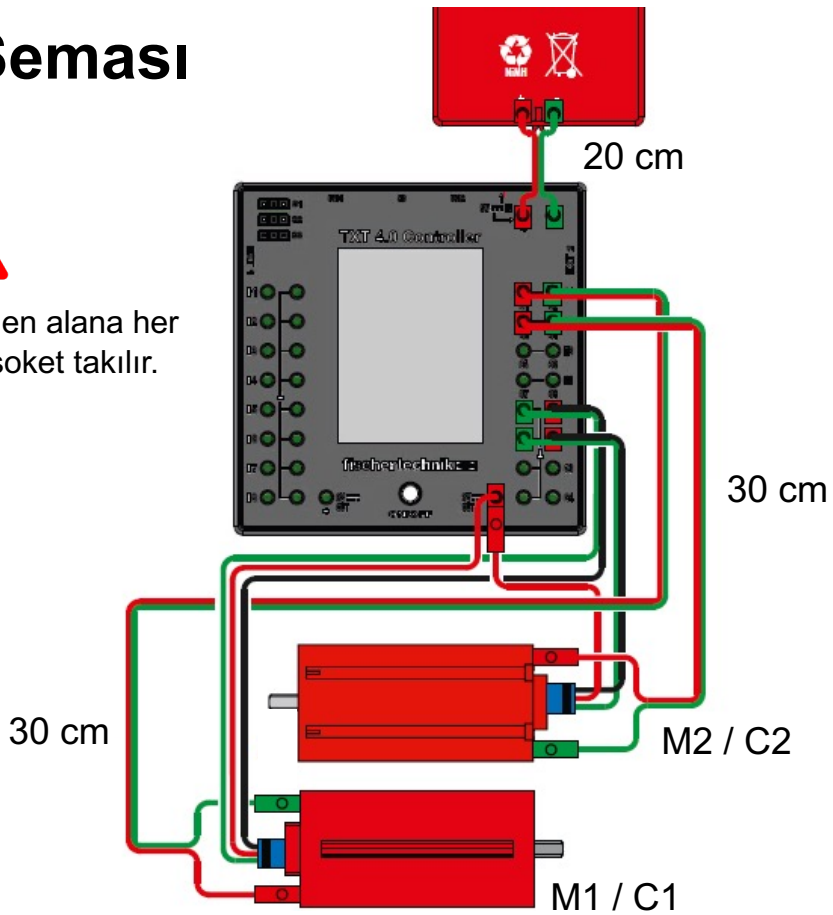
Model 8 – Buggy Mobil Robot





Devre Şeması

(+) ile gösterilen alana her zaman kırmızı soket takılır.



Buggy Mobil Robot Öğrenme Hedefleri

Konu

Üç tekerlekli mobil robot, alanda bağımsız bir şekilde hareket edebiliyor, engelleri ve sınır çizgilerini tespit edip bunlardan kaçınabiliyor.

İki motorun senkron kontrolü, engel algılama ve navigasyon.

Öğrenme hedefi

- Eşzamanlı motor kontrolünü anlamak
- Sensörler kullanılarak paralel süreçlerin senkronizasyonu
- Kodlayıcı darbeleri aracılığıyla gezinme

Gerekli malzemeler

- Program geliştirme için bilgisayar veya tablet.
- Programı TXT4.0'a iletmek için USB kablosu veya BLE veya WiFi bağlantısı.
- Siyah, 2 cm genişliğinde robot parkuru
- Engel (kutu, kitap, vb.)

Modelin kurulumu

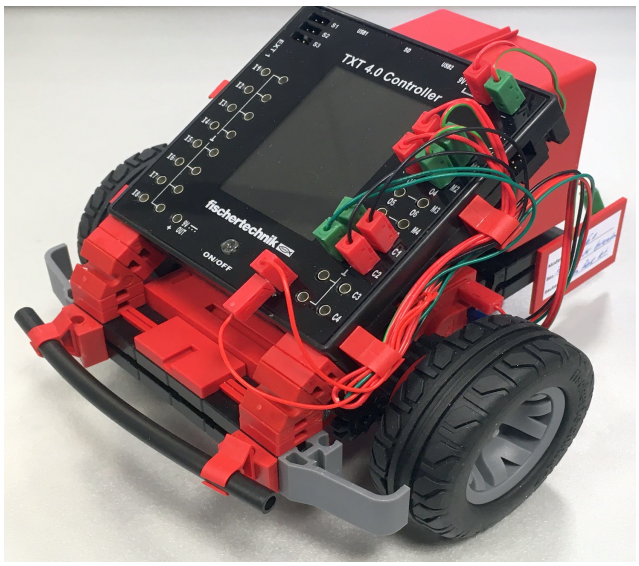
Küçük arabayı yapım talimatlarına göre yapın. İki motoru TXT'nin motor çıkışları olan M1'e (sol motorun hareket yönünde) ve M2'ye (sağ motor) bağlayın; kodlayıcılar C1'e (sol motor) ve C2'ye (sağ motor) bağlanır (devre şemasına bakın).

Daha fazla bilgi

[1] Durum diyagramları oluşturmak için çevrimiçi diyagram düzenleyici

(Format drawio): https://www.diagrammeditor.de/

Buggy Mobil Robot İçin Verilen Görevler



Buggy Mobil Robot

Programlama görevleri (GÖREV)

1. Basit ileri seyahat:
İki motoru ileri doğru hareket ettiren ve belirli bir süre sonra (örneğin beş saniye) tekrar durduran basit bir Blockly programı yazın.
Hızı kolayca değiştirebilmeniz için bir değişken kullanmalısınız.

2. Senkron sürücü:
Her iki motorda da aynı hız ayarlanmış olsa bile, arabanın tam olarak düz gitmediğini göreceksiniz. Bunun nedeni, motorların çıkışlarının ve yatakların sürtünme dirençlerinin asla tam olarak aynı olmamasıdır. Bu nedenle, tahrik eksenlerinin dönüş hızları aynı olmayacaktır.
Bunu, iki motorun (M1 ve M2) eksenlerini elektronik olarak senkronize ederek düzeltebilirsiniz. Programınızı, iki motorun senkronize olarak dönmesi için değiştirin.

3. Kodlayıcılarla kontrollü seyahat:
Daha önce yapılan modellerde öğrendiğiniz gibi, iki motorun manyetik kodlayıcıları saniyede 63,9 darbe verir. Bu darbeleri kullanarak arabanın belirtilen bir mesafe boyunca oldukça hassas bir şekilde gitmesini sağlayabilirsiniz.
Programınızı, arabanın belirli bir mesafe (örneğin 2 m) boyunca düz bir şekilde ilerlemesi için değiştirin. Bunu yapmak için bu mesafeye karşılık gelen darbe sayısını hesaplamanız gerekecektir.
İpucu: Hesaplama için lastiklerin çevresine ihtiyacınız olacak. Bunu ölçmenin en kolay yolu lastiğin etrafına bir kağıt şeridi yerleştirmek, çevreyi işaretlemek ve ardından şeridin işarete kadar olan uzunluğunu ölçmektir.
Ölçümünüzü (test) programınızla kontrol edin ve dönüştürme faktörünü düzeltin (gerekirse).

Lütfen dikkat: Programın bitmesinden önce motorların belirlenen darbe sayısına ulaşmasını beklemelisiniz.

Buggy Mobil Robot İçin Verilen Görevler

4. Yerinde senkronize dönüş:

Şimdi, buggy yerinde dönmeyi öğrenmelidir. Bunu yapmak için, önce 250 ms boyunca sağa dönmeli, sonra aynı sayıda impuls boyunca sola geri dönmelidir. Devirleri beş kez tekrarlayın; sonra buggy program başladığında olduğu gibi hizalanmalıdır.

İpucu: Motorlar durduktan sonra biraz dönmeye devam edecektir. Arabanın başlangıç noktasına geri dönmesi için bu sayıda darbe kadar geri dönmeniz gerekecektir.

5. Belirli bir açı etrafında dönme:

Şimdi, arabanın (yerinde) etrafında dönmesi gereken bir açı belirtmek istiyoruz. Bunu yapmak için, öncelikle motorların dönüş açısı derecesi başına kaç darbe dönmesi gerektiğini hesaplamalısınız.
Robotu kendi ekseni etrafında birkaç kez döndüren basit bir test programı kullanarak hesaplamanızın doğru olduğundan emin olun ve gerekirse hesaplanan dönüşüm faktörünü düzeltin.
Robotun dönmesine ve ardından yerine geri dönmesine neden olan kısa bir program yazın – 45°, 90°, 180° ve 360°.

6. Kodlayıcılarla kontrollü seyahat:

Daha önce yapılan modellerde öğrendiğiniz gibi, iki motorun manyetik kodlayıcıları saniyede 63,9 darbe verir. Bu darbeleri kullanarak arabanın belirtilen bir mesafe boyunca oldukça hassas bir şekilde gitmesini sağlayabilirsiniz.
Programınızı, arabanın belirli bir mesafe (örneğin 2 m) boyunca düz bir şekilde ilerlemesi için değiştirin. Bunu yapmak için bu mesafeye karşılık gelen darbe sayısını hesaplamanız gerekecektir.
İpucu: Hesaplama için lastiklerin çevresine ihtiyacınız olacak. Bunu ölçmenin en kolay yolu lastiğin etrafına bir kağıt şeridi yerleştirmek, çevreyi işaretlemek ve ardından şeridin işarete kadar olan uzunluğunu ölçmektir.
Ölçümünüzü (test) programınızla kontrol edin ve dönüştürme faktörünü düzeltin (gerekirse).

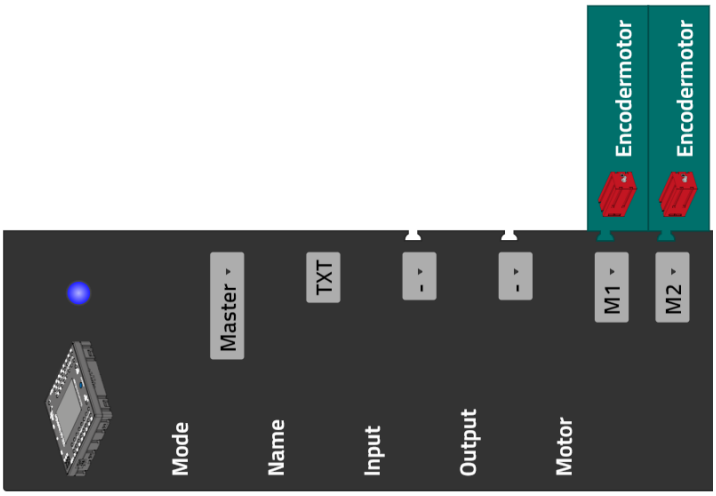
Lütfen dikkat: Programın bitmesinden önce motorların belirlenen darbe sayısına ulaşmasını beklemelisiniz.

Buggy Mobil Robot Verilen Çözümler

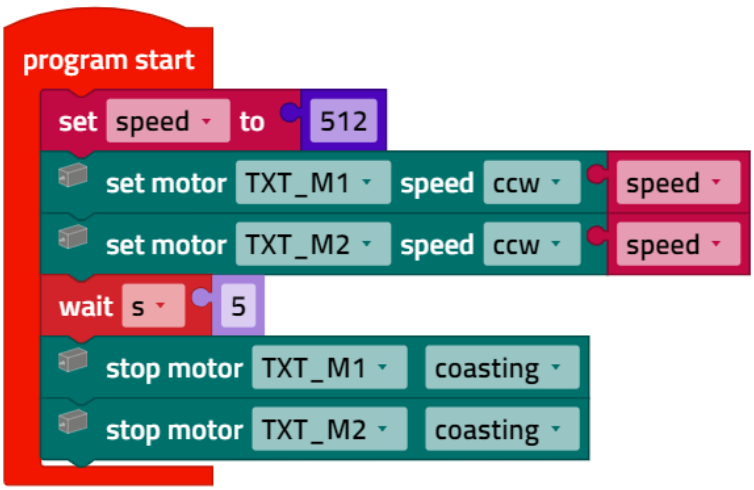
Bu programlama görevindeki 6 alt görev, öğrencilere kodlayıcı motorlarının senkron kontrolünü adım adım tanıtıyor ve doğrudan birbirinin üzerine inşa ediyor.

Programlama görevleri (ÇÖZÜM)

1. Basit ileri seyahat:
Motorların yapılandırılması:



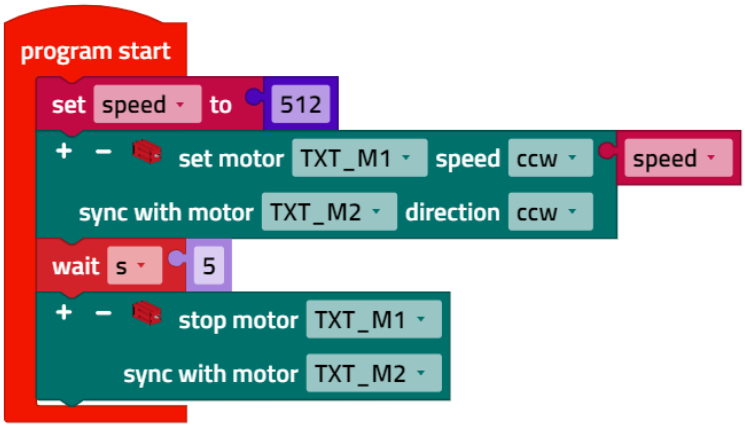
Program (örnek):



Buggy_Driving_Test.ft

Programlama görevleri (ÇÖZÜM)

2. Senkron sürücü:
Program (örnek):

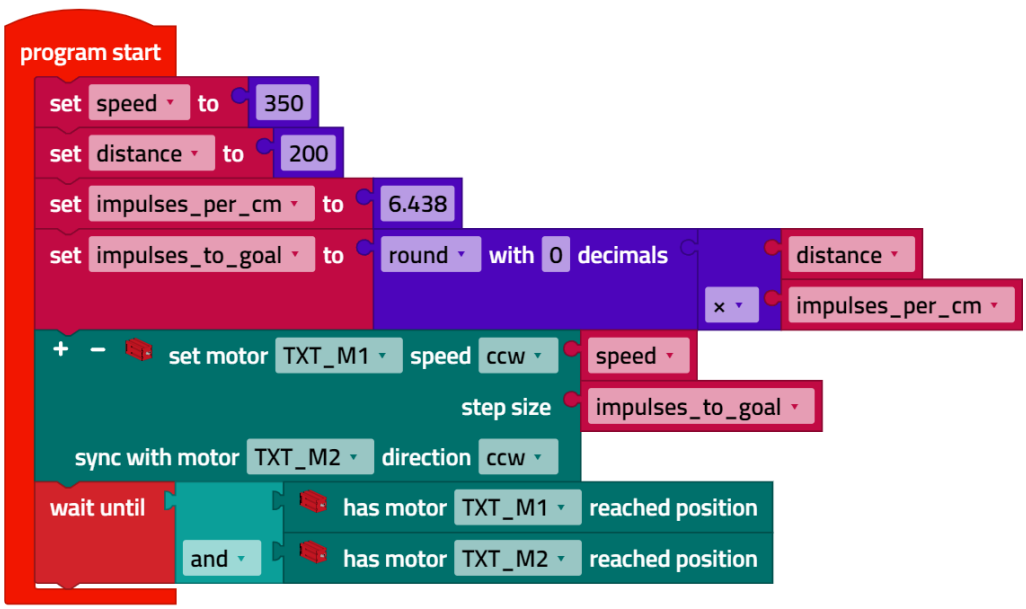


Buggy_Driving_Test_Synchronous.ft

3. Kodlayıcılarla kontrollü seyahat
Arabanın lastiklerinin çevresi yaklaşık 20,5 cm'dir. Bu nedenle, tekerlekler 1:2 dişli oranına sahip motorlar tarafından düşürüldüğünden, kodlayıcı $63.9 \cdot 2 \cdot \frac{100}{20.5} \approx 623$ darbe/m

(veya 6,23 darbe/cm) verir. İki metrelik bir mesafede yapılan testler, değerin yaklaşık **6,438 darbe**/cm'ye düzeltilmesi gerektiğini göstermektedir (bu değer modelden modele farklılık gösterebilir). Bu nedenle, 2 m ileri gitmek için 1288 darbe gerekir.

Program (örnek):

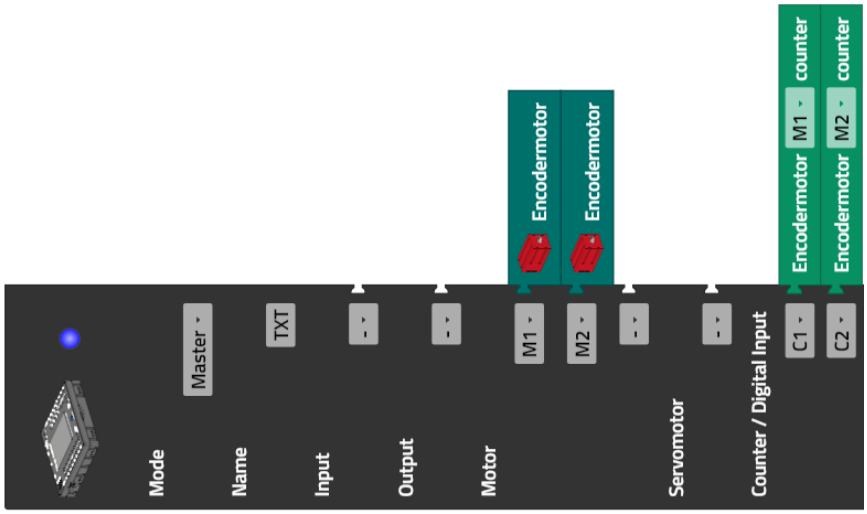


Buggy_Driving_Test_Synchronous_Distance.ft

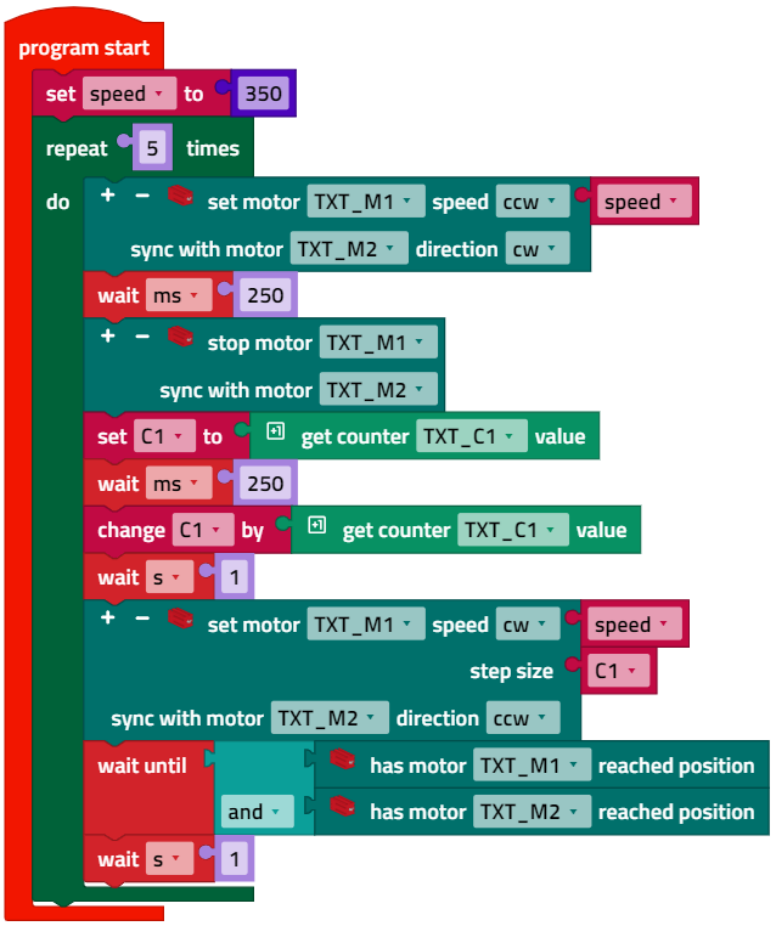
Programlama görevleri (ÇÖZÜM)

4. Yerinde senkronize dönüş

Kodlayıcının (sayacın) yapılandırılması:



Program (örnek):

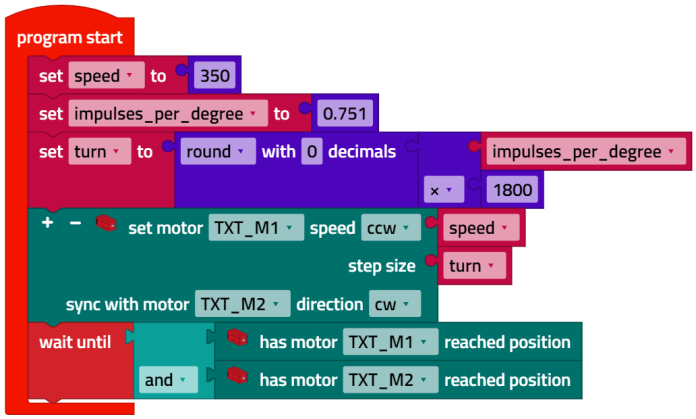


Buggy_Turning_Test_Synchronous.ft

5. Yerinde Belirli bir açı etrafında dönme

Yerinde dönerken lastikler, lastiğin merkezine göre yaklaşık 14 cm çapında bir daire tamamlar. Bu nedenle, dairenin çevresi etrafında seyahat etmek yaklaşık $14 \cdot \pi \cdot 6.132 \approx 269.7$ darbeye veya $\frac{269.7}{360} \approx 0.749$ darbe/dereceye karşılık gelir. Bu değer yalnızca kaba bir tahmindir, çünkü lastikler yapıları gereği tam olarak merkezde yuvarlanmazlar. Değer, örneğin robotun kendi ekseni etrafında beş kez dönmesini sağlayan basit bir program kullanılarak deneyerek kontrol edilebilir.

Dönüşüm faktörünü (5 dönüş) test etmek için program (örnek):

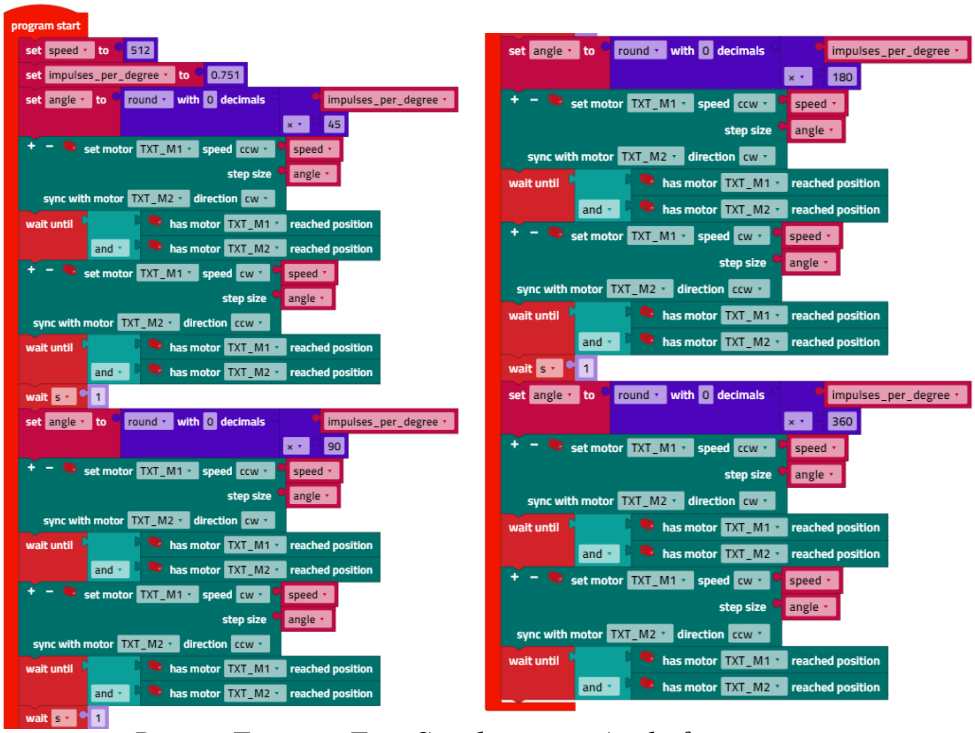


Buggy_Synchronous_Turning_Calibration.ft

Yapılan testler dönüşüm faktörünün **0,751 darbe/derece** civarına ayarlanması gerektiğini göstermektedir (bu değer modelden modele farklılık gösterebilir).

Enkoder motorlarının ters yönde çalışmasıyla 90° dönüş 68 darbe gerektirirken, 45° dönüş yaklaşık 34 darbe, 180° dönüş 135 darbe ve tam dönüş ise 270 darbe gerektirir.

Program (örnek):



Buggy_Turning_Test_Synchronous_Angle.ft

ftrobotik

fischertechnik

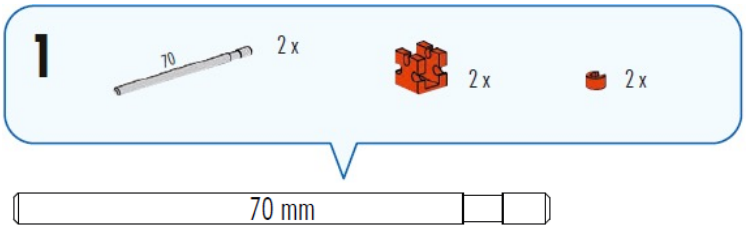
BÖLÜM 04

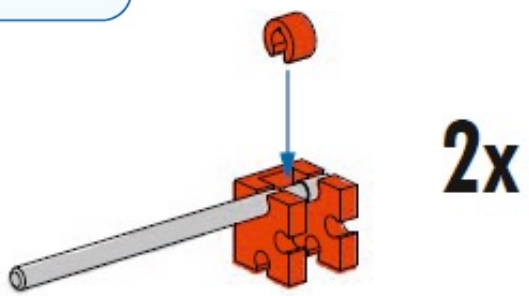
Çizim Robotu

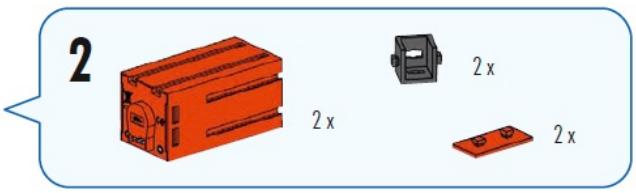
Öğrenme hedefi

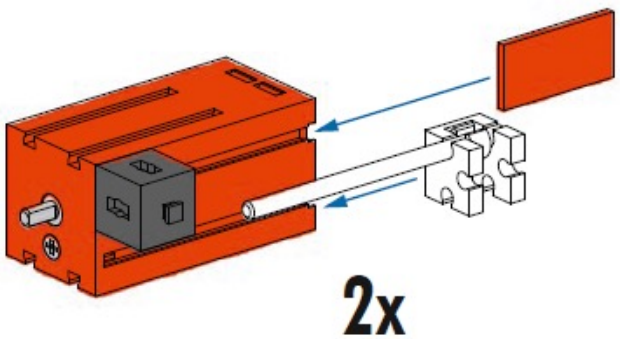
- ✓ Net program tasarımı için fonksiyonların kullanımı
- ✓ Motor darbeleri aracılığıyla navigasyon
- ✓ Seyahat davranışını modelleme ve hesaplama

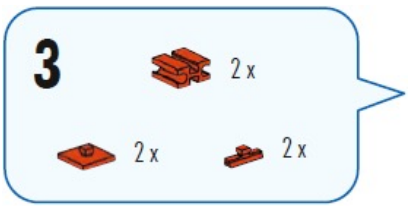
MODEL 11 Çizim Robotu

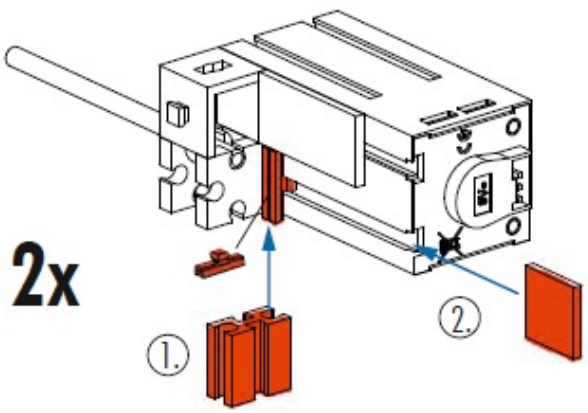


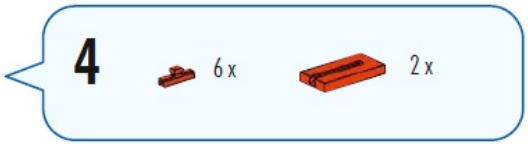


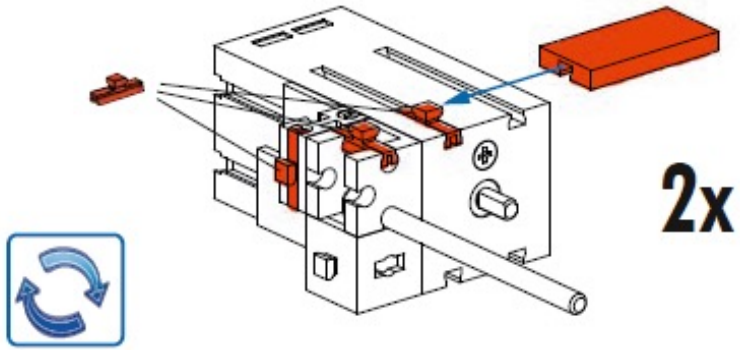


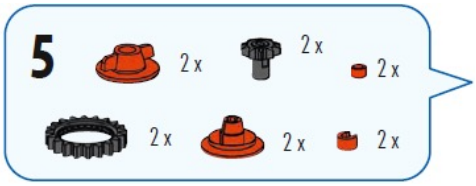


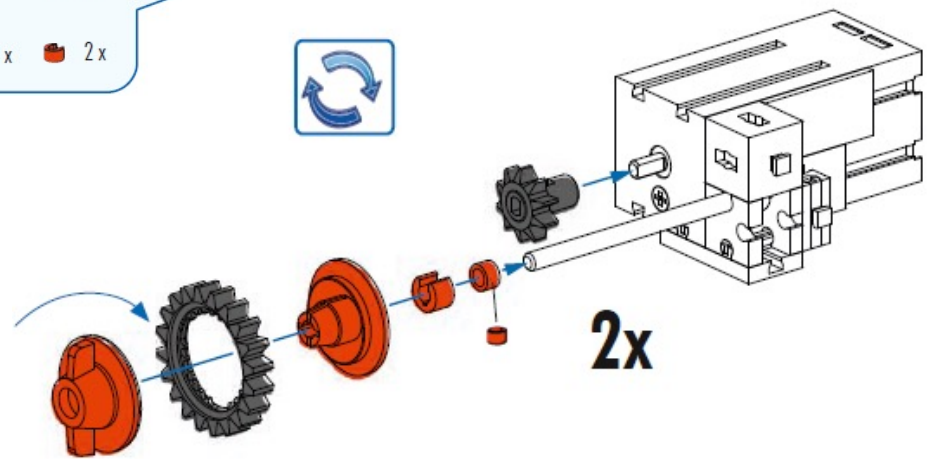


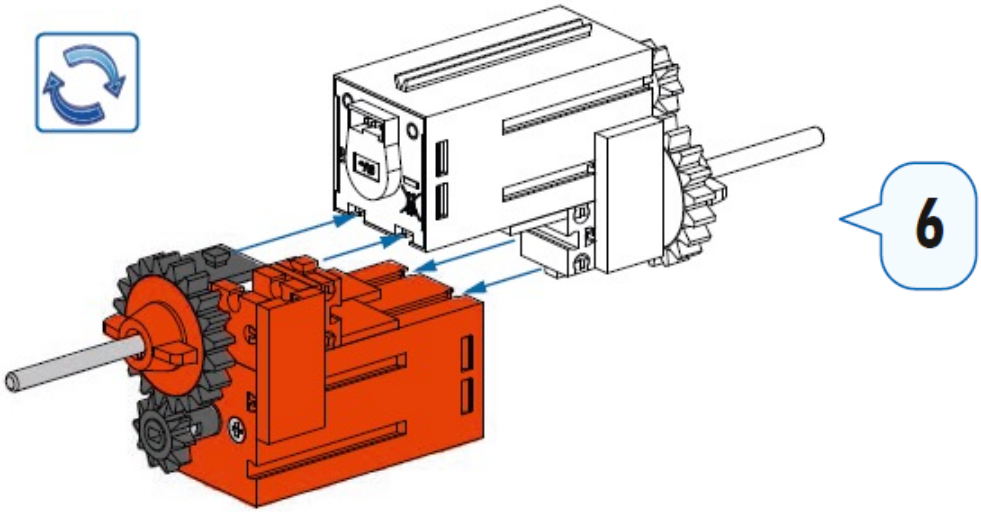


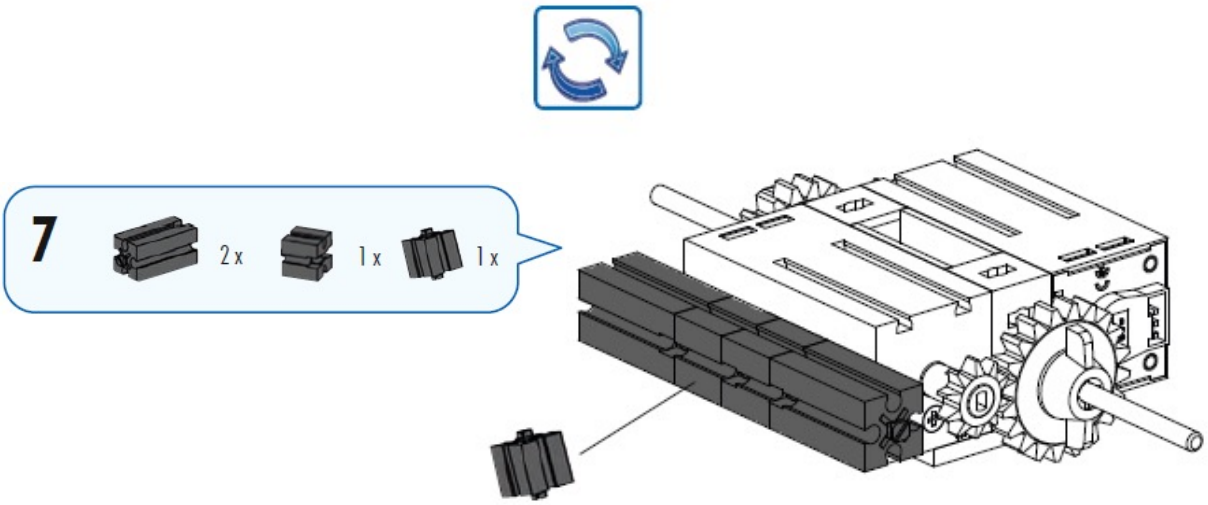

4
6 x
2 x

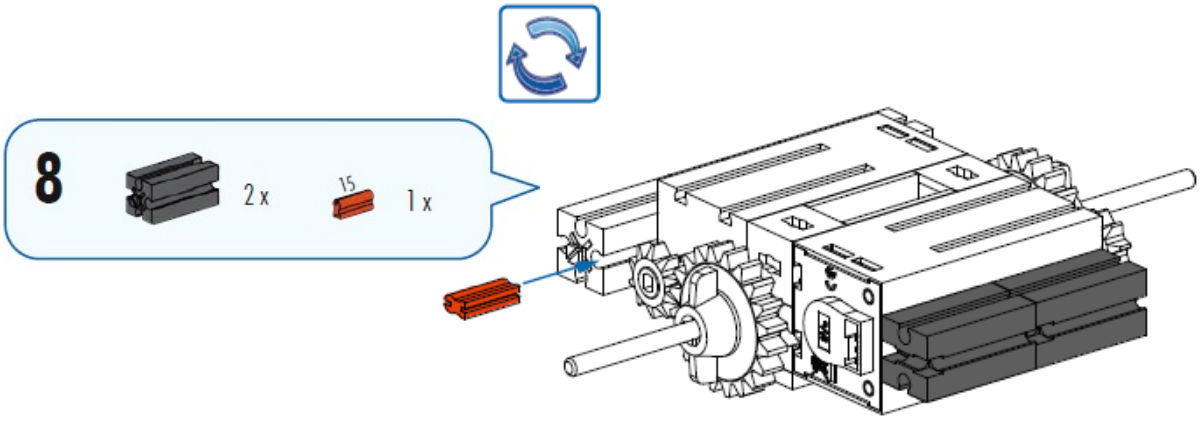

2x

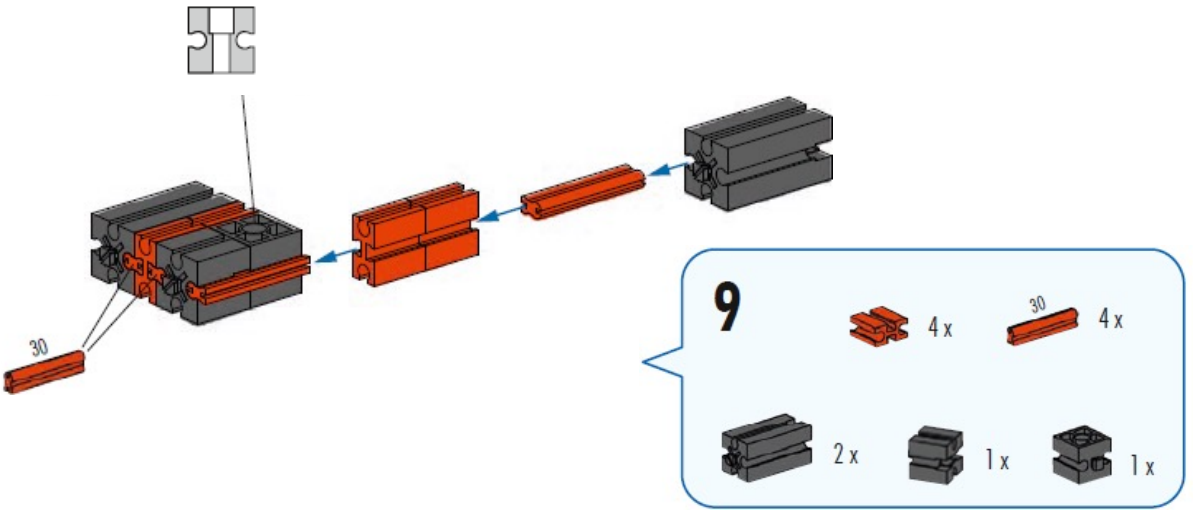

5
2 x
2 x
2 x
2 x
2 x
2 x

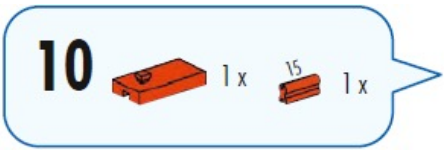

2x

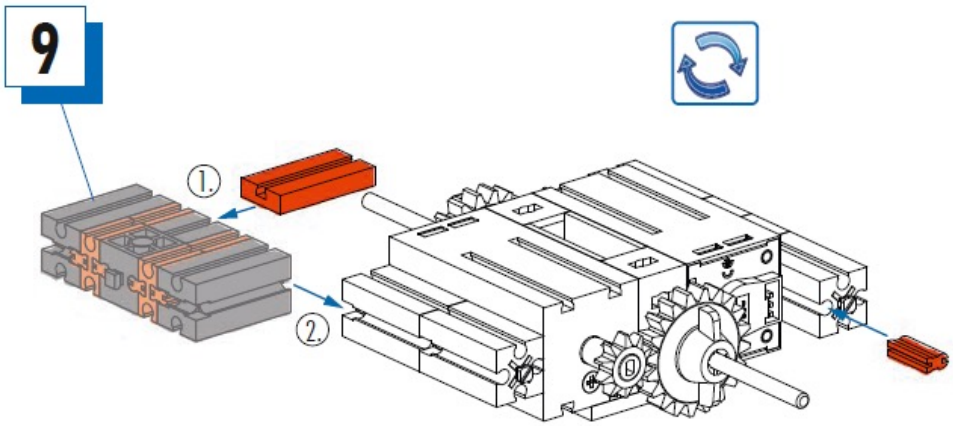

6

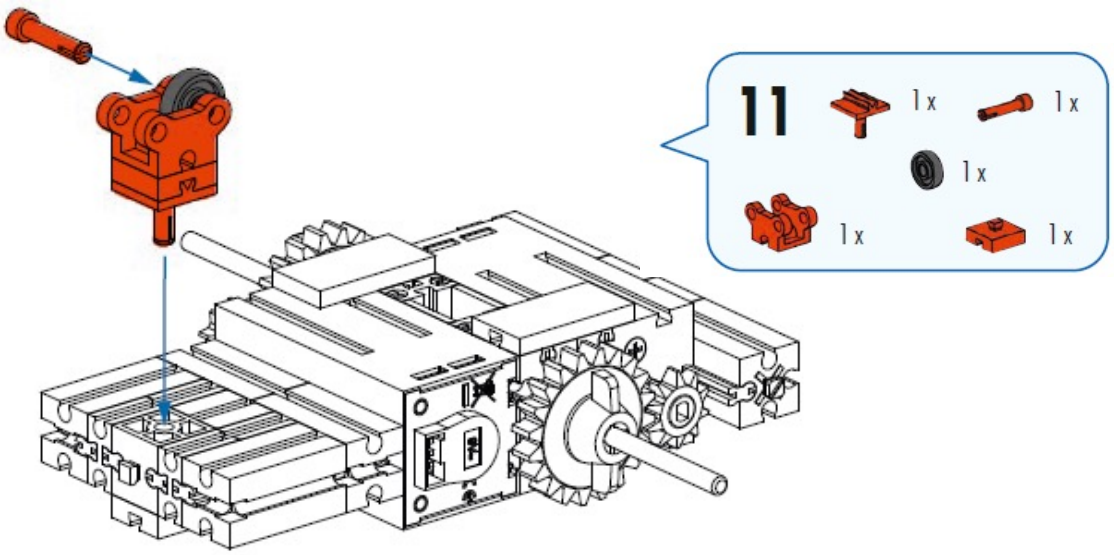

7
2 x
1 x
1 x

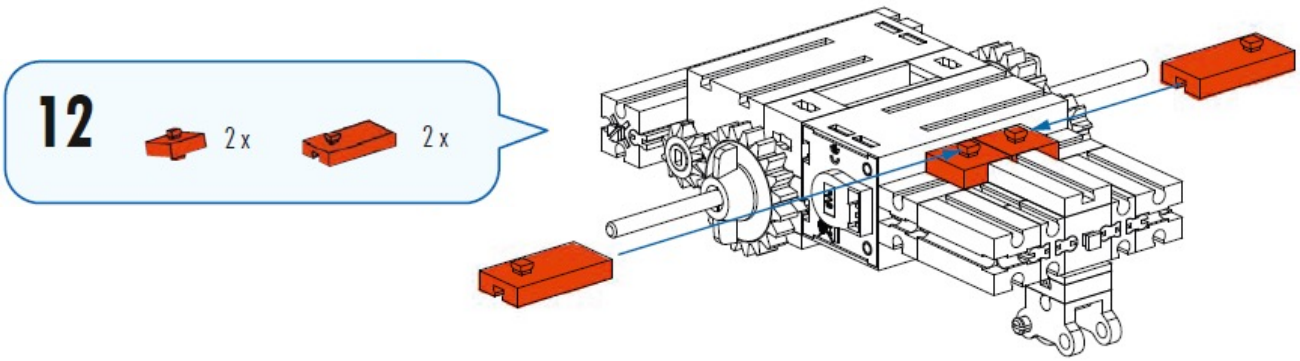

8
2 x
15
1 x

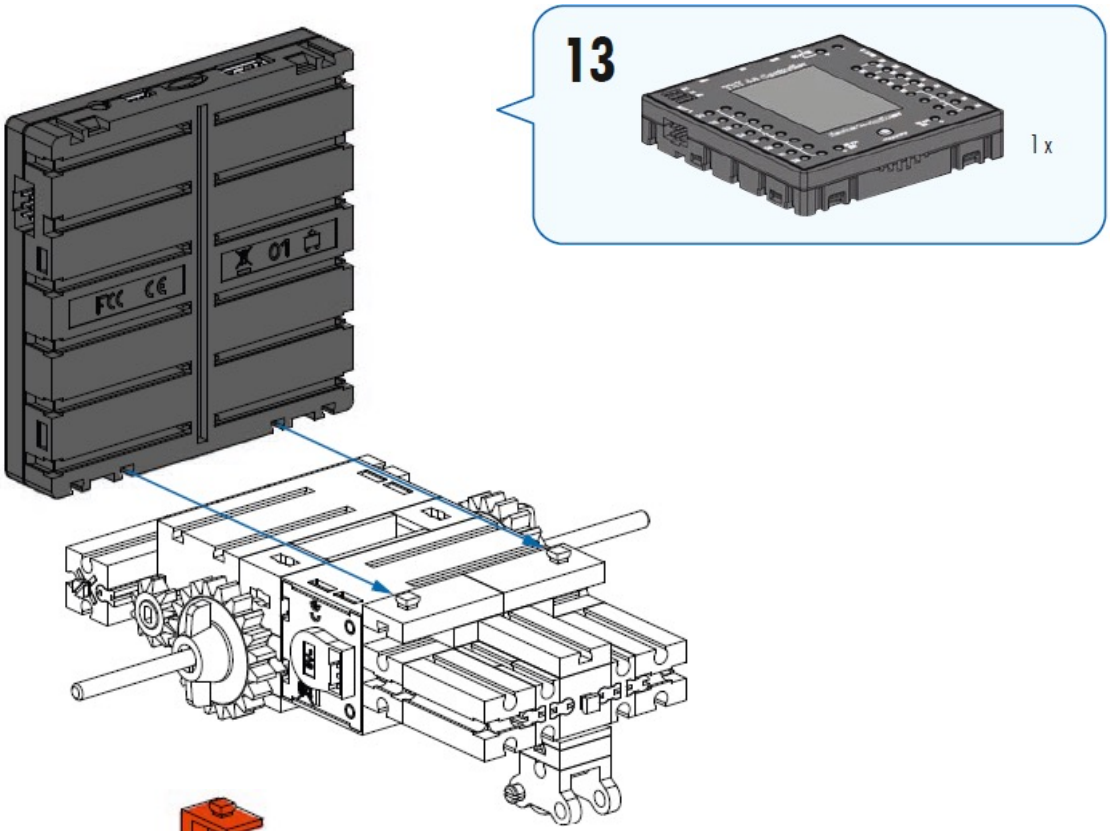

30
9
4 x
30
4 x
2 x
1 x
1 x

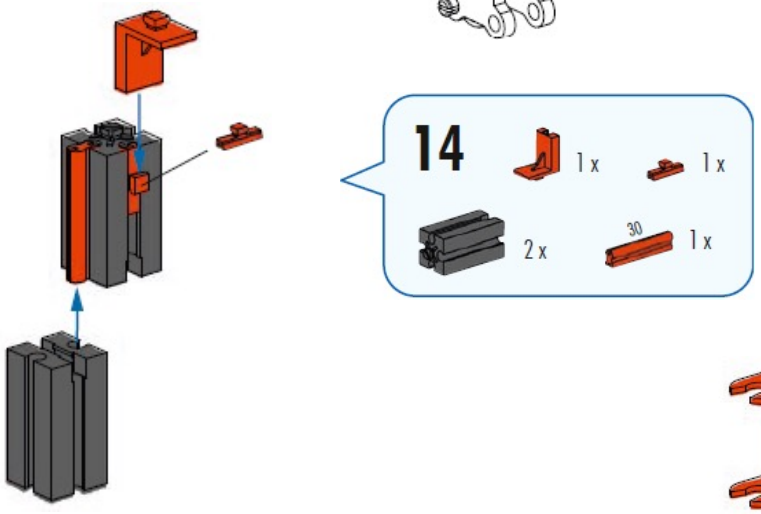

10
1 x
15
1 x

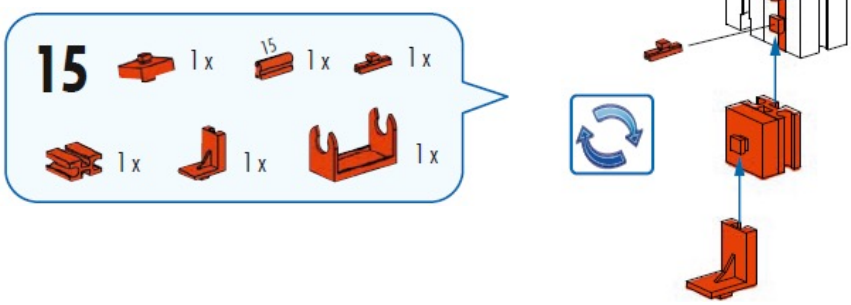

9
1.
2.

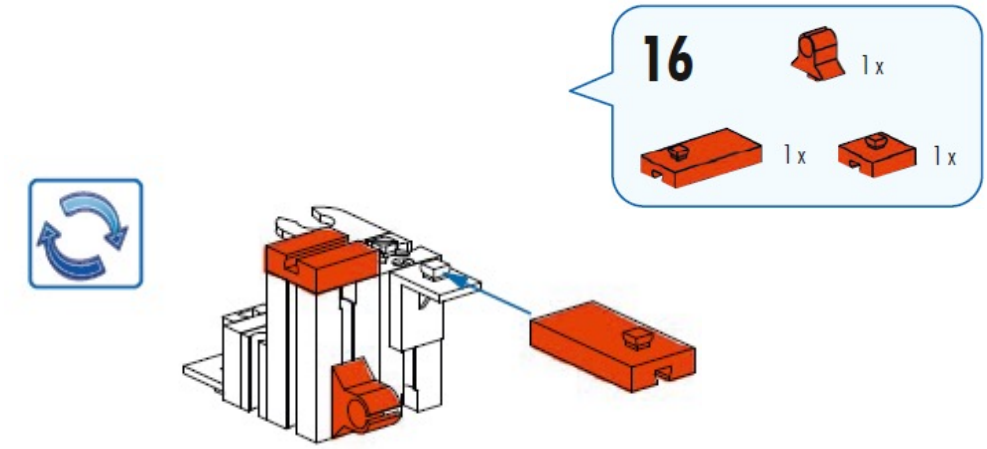

11
1 x
1 x
1 x
1 x
1 x

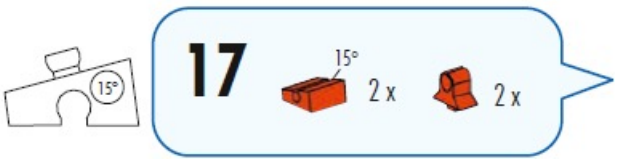

12
2 x
2 x

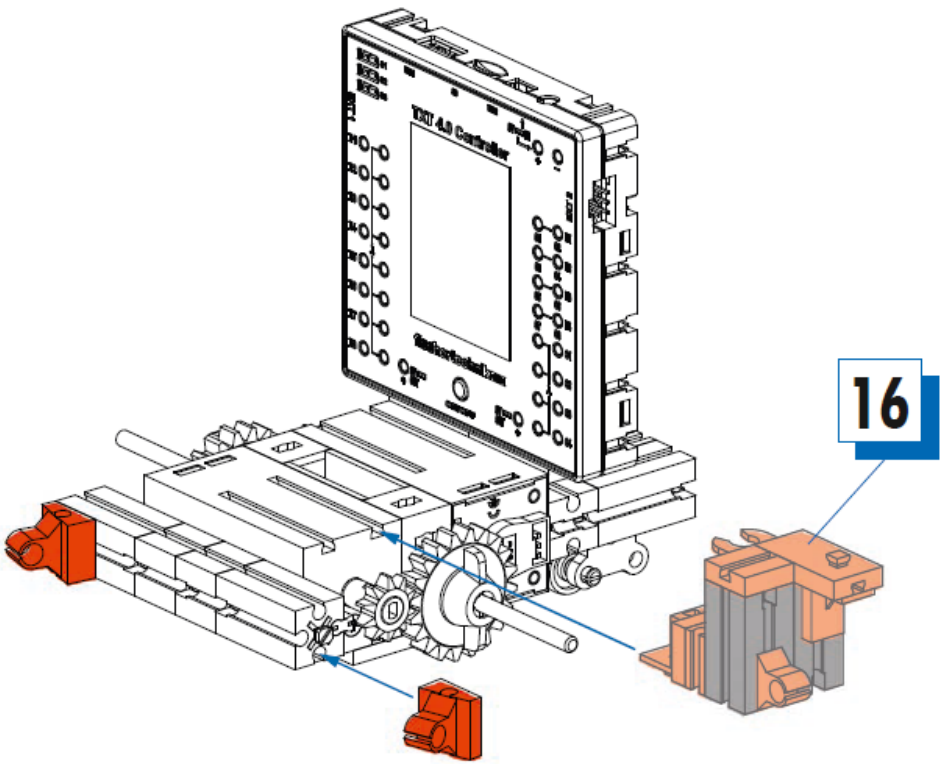

13
1 x


14
1 x
1 x
2 x
30
1 x

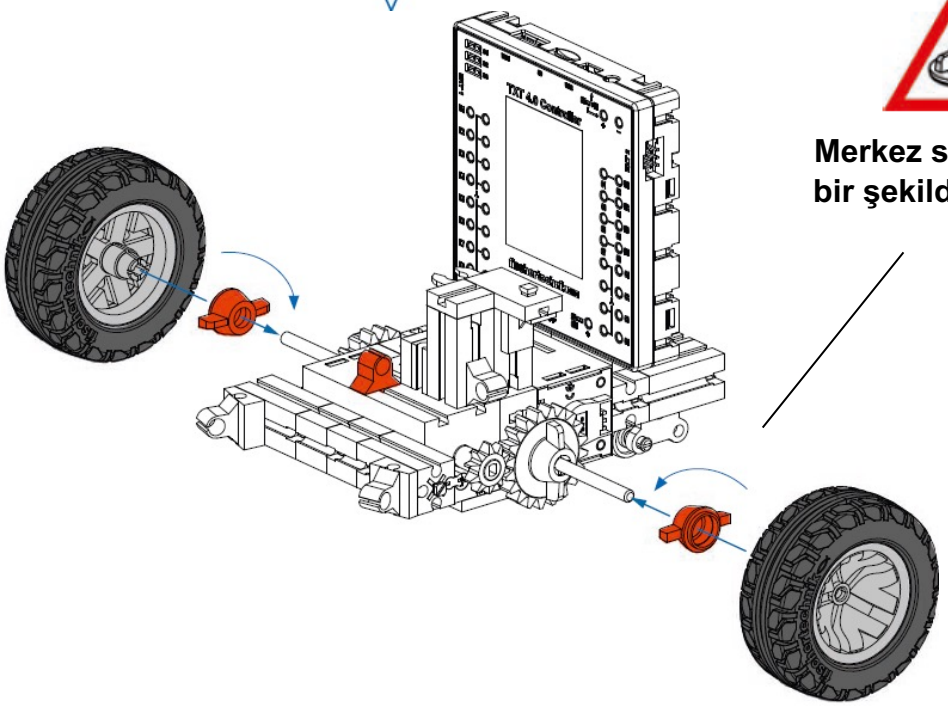

15
1 x
15
1 x
1 x
1 x
1 x
1 x

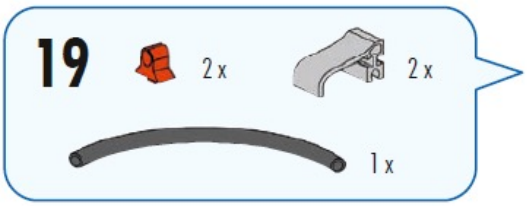

16
1 x
1 x
1 x

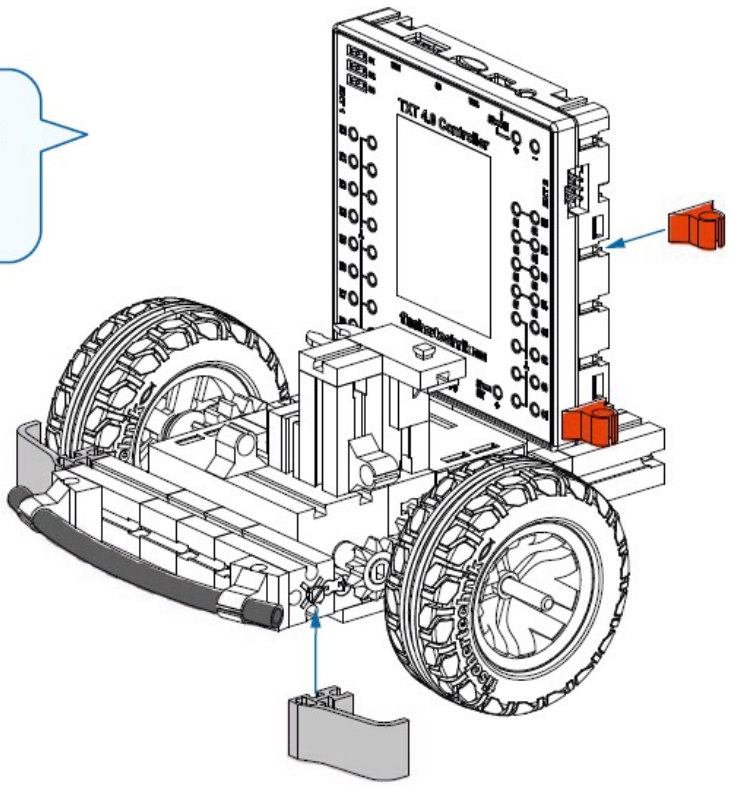

15°
17
15°
2 x
2 x

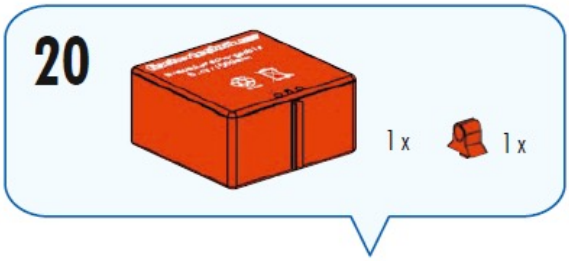

16

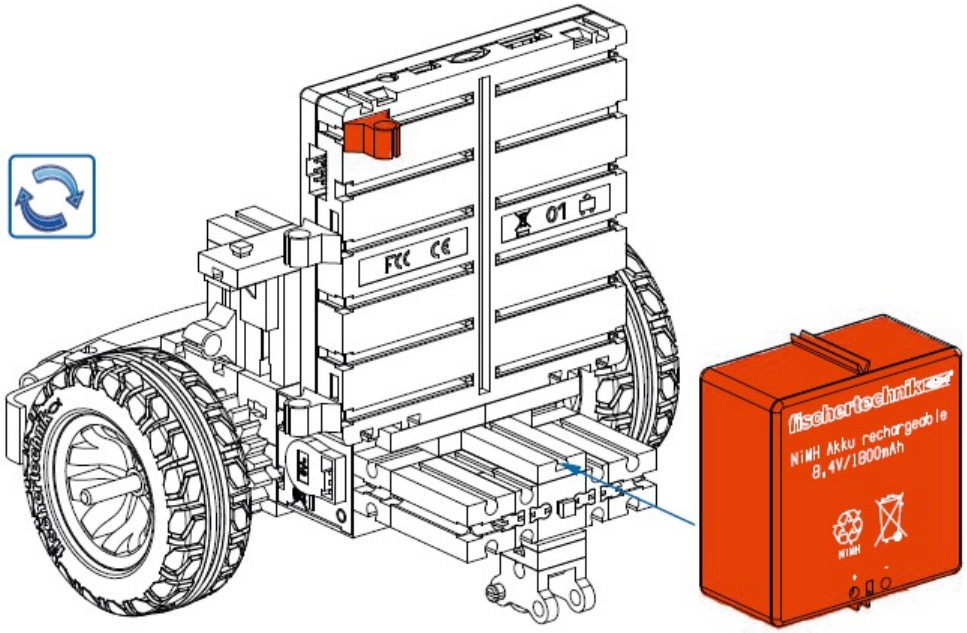
Merkez somunu sıkı bir şekilde ayarlayın

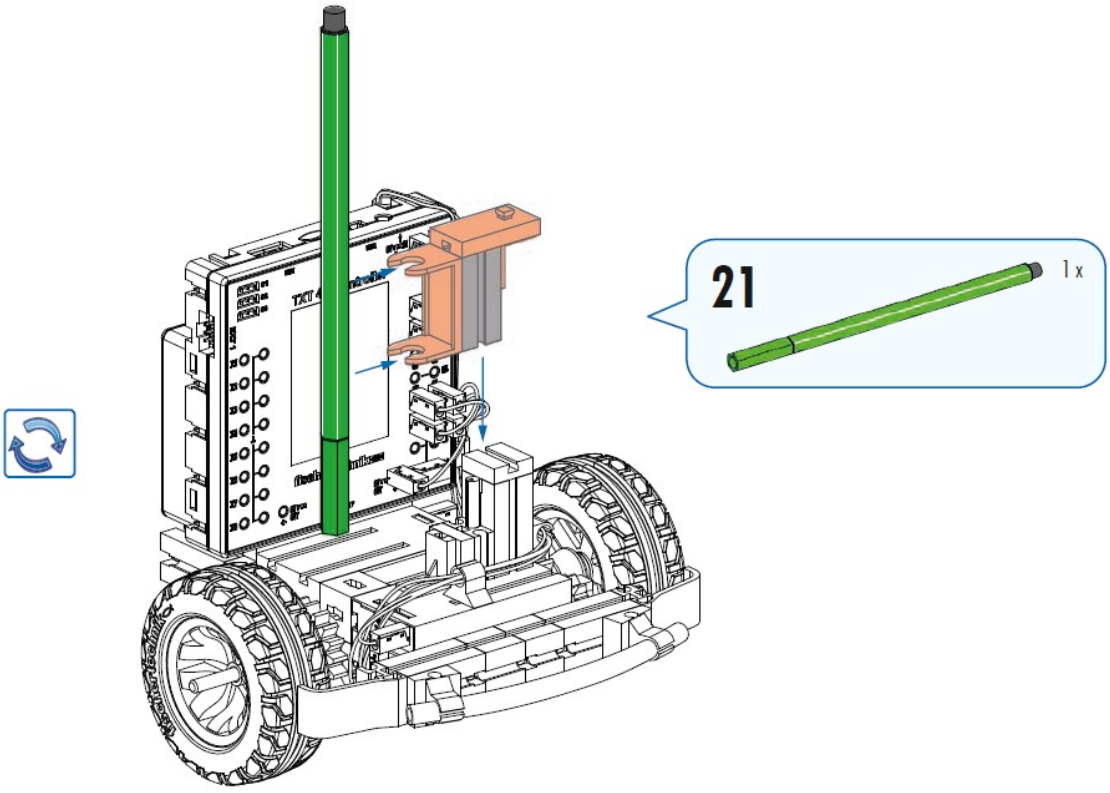


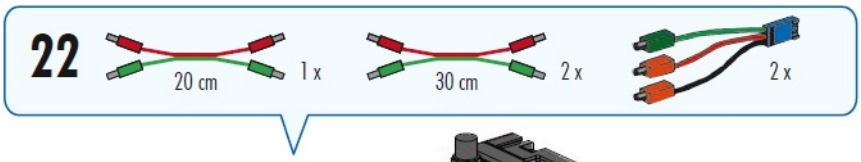




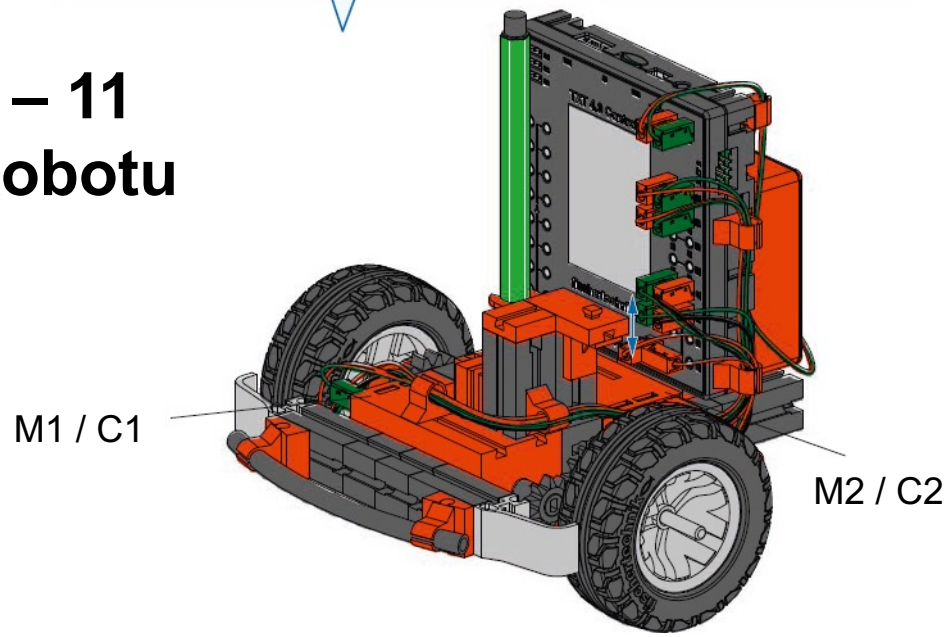

20
1x
1x


fischertechnik
NiMH Akku rechargeable
8,4V/1900mAh

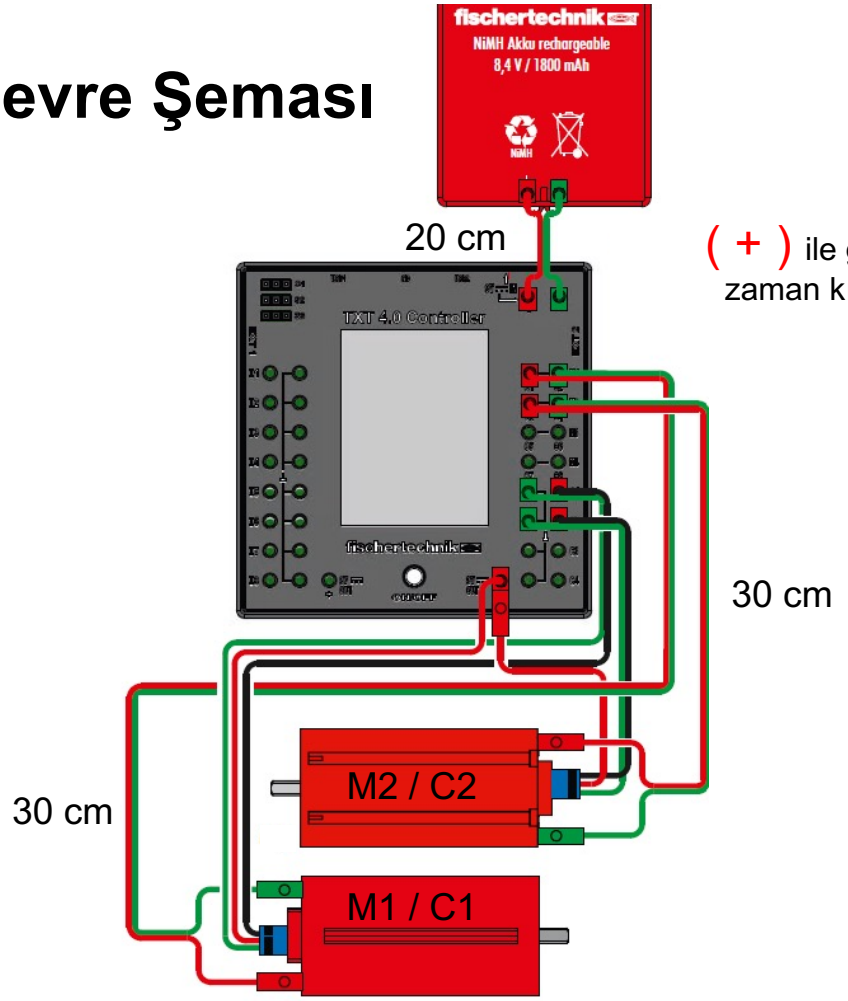

21
1x



Model – 11
Çizim Robotu



Devre Şeması



(+) ile gösterilen alana her zaman kırmızı soket takılır.

Çizim Robotu Öğrenme Hedefleri

Konu

Artık buggy modelimiz bir çizim robotuna dönüşecek: Bir işaretleyici veya ince uçlu kalem kullanarak, belirtilen özelliklere göre geometrik şekiller (çizgiler, köşeler, daireler) çizecek.

Geometrik şekillerin çizilmesinde iki motorun hassas kontrolü.

Öğrenme hedefi

- Net program tasarımı için fonksiyonların kullanımı
- Motor darbeleri aracılığıyla navigasyon
- Seyahat davranışını modelleme ve hesaplama

Gerekli malzemeler

- Program geliştirme için bilgisayar veya tablet.
- Programı TXT4.0'a iletmek için USB kablosu veya BLE veya WiFi bağlantısı.
- Kalem (ince uçlu kalem, keçeli kalem), büyük bir beyaz kağıt
- Program şablonu “Drawing_Coordinates.ft”

Modelin kurulumu

Çizim robotunu yapım talimatlarına göre inşa edin veya arabanın temel modelini çizim robotuna dönüştürün. Aktüatörler için, sadece motor çıkışları M1'deki (sol motorun hareket yönünde) ve M2'deki (sağ motor) iki motora ve ayrıca C1'deki (sol motor) ve C2'deki (sağ motor) kodlayıcıya ihtiyacınız var. Dikey tutucuya bir pim sabitleyin (şekle bakın).

Daha fazla bilgi

[1] Search for “Coordinate Grid Picture” online.

[2] Oliver Boorman: *Cartesian Grid Image Generator.*

Çizim Robotu İçin Verilen Görevler

Çizim Robotu

Programlama görevleri (GÖREV)

1. Kontrol fonksiyonları:

Model 6'da, arabaya tekerleklerini senkronize bir şekilde nasıl hareket ettireceğini öğrettiniz (düz ileri sürme, dönme). Çizim robotu için, başlamak için sadece iki temel harekete ihtiyacınız var:

- (eşzamanlı) belirli bir mesafe boyunca ileri seyahat ("ileri"), bir çizgi çizmek için manyetik kodlayıcının darbe sayısına göre hesaplanır ve
- Aracın kendi ekseni etrafında belirli bir açıyla dönmesi (“turn_left” ve “turn_right”), aynı zamanda manyetik kodlayıcı için darbelere dönüştürülür.

Bu temel hareketleri tamamlamak için, yalnızca seyahat mesafesini cm cinsinden veya dönüş açısını derece cinsinden iletmeniz gereken iki işlev geliştirin. Bir sonraki kontrol komutunu yürütmeden önce motorların önce durması gerektiğini unutmayın.

Çizim robotunuzu kullanarak fonksiyonları test edin.

2. Çapraz ev bulmacası:

Bu iki işlevle artık çizim robotunuzu geometrik şekiller çizmek için kullanabileceğiniz "araçlara" sahipsiniz. Basit bir görevle başlayacağız: Arabanızı, "çapraz ev bulmacasını" kesintisiz bir sırayla çizebilecek şekilde programlayın.

3. N- kenarlı çokgen:

Şimdi, çizim robotu sabit kenar uzunluğuna sahip herhangi sayıda kenarı olan bir çokgen çizebilmelidir. Programı olabildiğince genel ve - döngüler kullanarak - olabildiğince kompakt olacak şekilde tasarlamaya çalışın.
Robota bir üçgen, sonra bir kare, sonra bir beşgen ve en sonunda 15 kenarlı bir çokgen çizmesini söyleyerek programı test edin.
İpucu: Kenar uzunluğunun çok uzun olmamasına dikkat edin.

Buggy Mobil Robot İçin Verilen Görevler

Deneysel görevler (GÖREV)

1. Hedefe doğru sürün:
Şimdi, çizim robotu koordinatlar biçiminde belirtilen bir (hedef) noktaya nasıl hareket edeceğini öğrenecek.

1a. Arabanızın (x,y) konumundan belirli bir noktaya hareket etmesini söylemek için neyi hesaplamanız gerekir? Önce bir çizim yapın.
1b. Arabayı belirtilen bir noktaya hareket ettiren bir program yazın.

2. "Sayılarla boyama"
Program şablonunda x ve y koordinatlarına sahip iki liste var
"Drawing_Coordinates.ft"

Buggy'niz için, belirtilen noktalara (koordinatlara) sırayla yaklaşan bir kontrol programı yazın. Başlangıç noktası, kağıdın merkezidir (koordinatlar (0,0)) ve yönü 0°'dir (x ekseni boyunca). Çizim kağıdı en az 60 cm genişliğinde ve 40 cm yüksekliğinde olmalıdır.
İsterseniz, buggy'nin tamamlaması için bir çizim için kendi koordinat listenizi belirtebilirsiniz.

3. Kesinlik:
Arabanın çizdiği nesnelerin hassasiyetini nasıl artırabilirsiniz?

Çizim Robotu Verilen Çözümler

Görevler, açık, kompakt ve anlaşılması kolay bir program oluşturmak için döngüler ve işlevler kullanma alıştırmasıdır. Öğrencilerin sonuçları, bu özellikleri karşılayıp karşılamadıklarını belirlemek için birbirleriyle karşılaştırılmalıdır. Deneysel görev (görev 6'daki deneysel göreve benzer) aynı şekilde bir modelleme alıştırmasıdır: Öğrenciler, uygun modellemenin daha basit ve daha açık programlarla sonuçlandığını öğreneceklerdir.

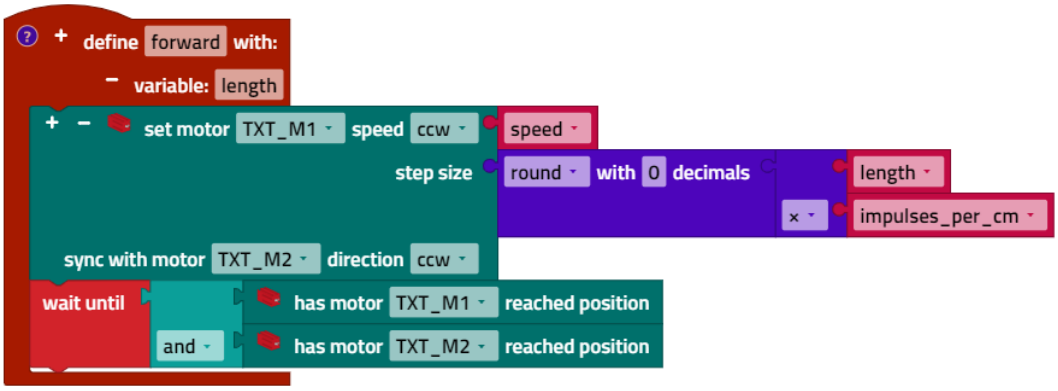
Programlama görevleri (ÇÖZÜM)

1. Kontrol fonksiyonları:

"d mesafesi boyunca ileri seyahat" ve "α açısıyla dönüş" fonksiyonları:

- Görev 6'da, bir mesafeyi darbelere dönüştürmek için yapılan deneylerle 6,438 darbe/cm'lik bir faktör belirlendi. Bu değer modelden modele biraz farklılık gösterebilir.

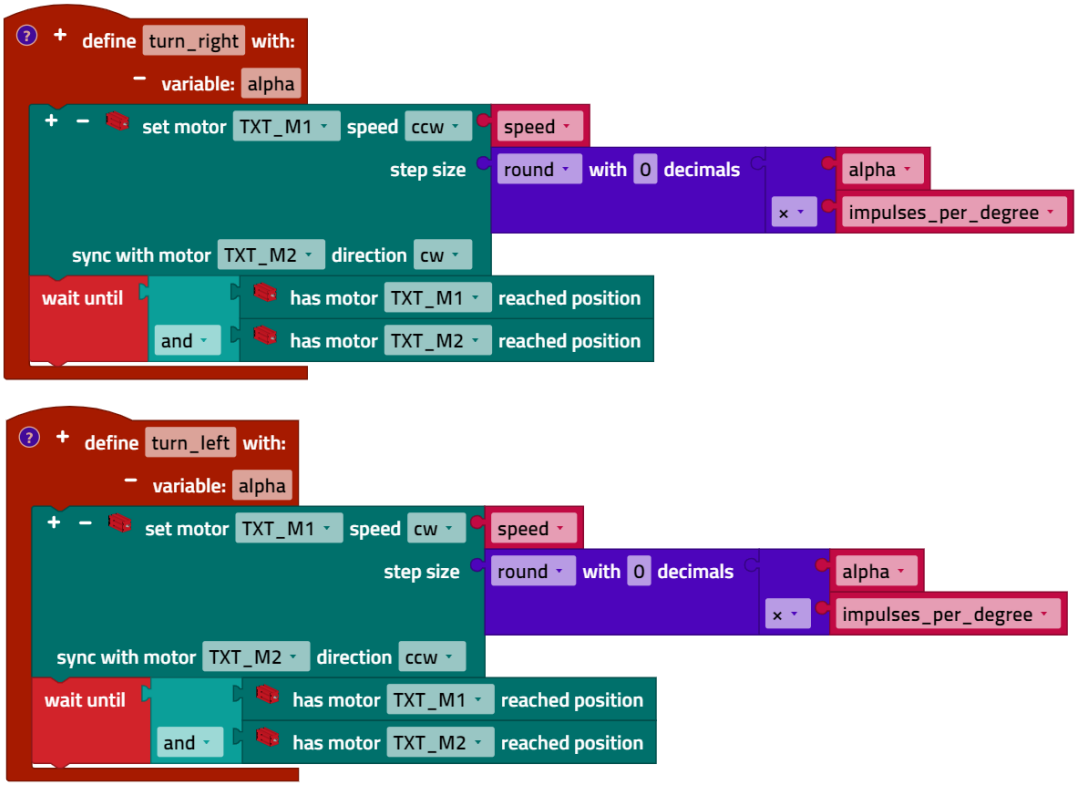
Program özeti (örnek):



Buggy_Driving_Functions.ft

- Yerinde dönüş için deneyler yoluyla 0,751 dürtü/derece dönüşüm faktörü belirlendi. Bu değer modelden modele de biraz farklılık gösterebilir. Aşağıdaki görevler için, sağa (saat yönünde) ve sola dönüş için ayrı bir işlev oluşturmak faydalı olacaktır.

Program (örnek):

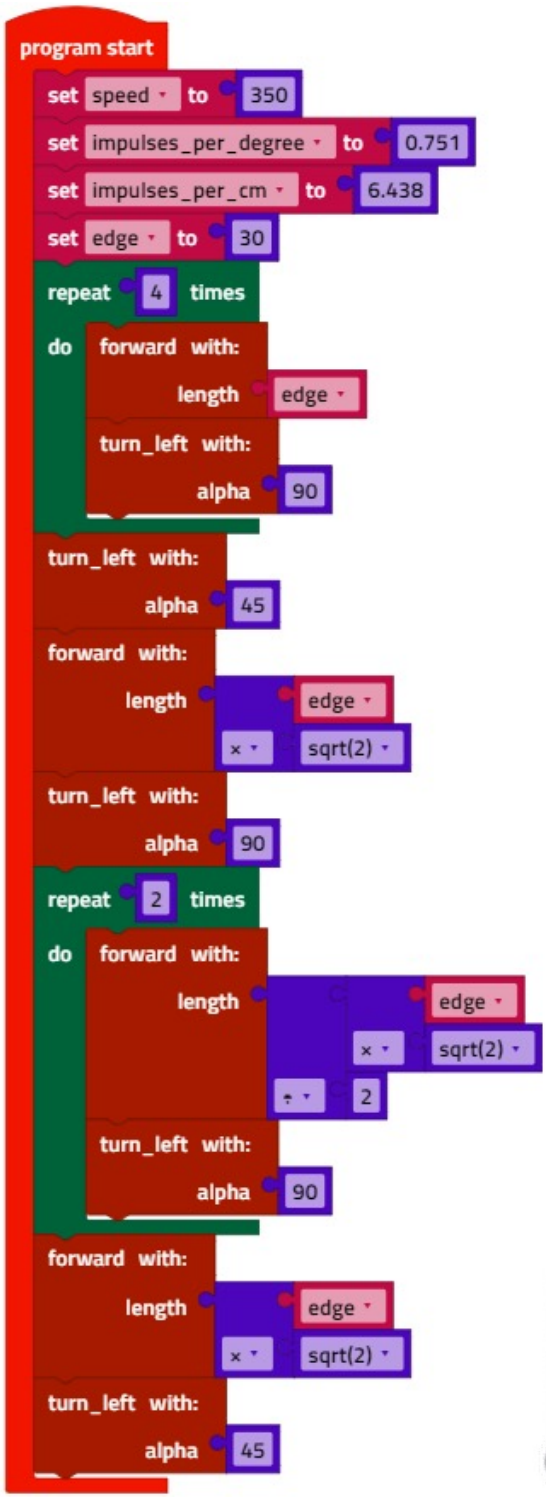


Buggy_Driving_Functions.ft

4. Çapraz ev bulmacası:

Eğri ev bulmacası, kenar uzunluğu 10 cm olan sekiz çizgide örneğin şu şekilde çizilebilir: 88 farklı doğru çözüm olasılığı vardır.

Program (örnek):

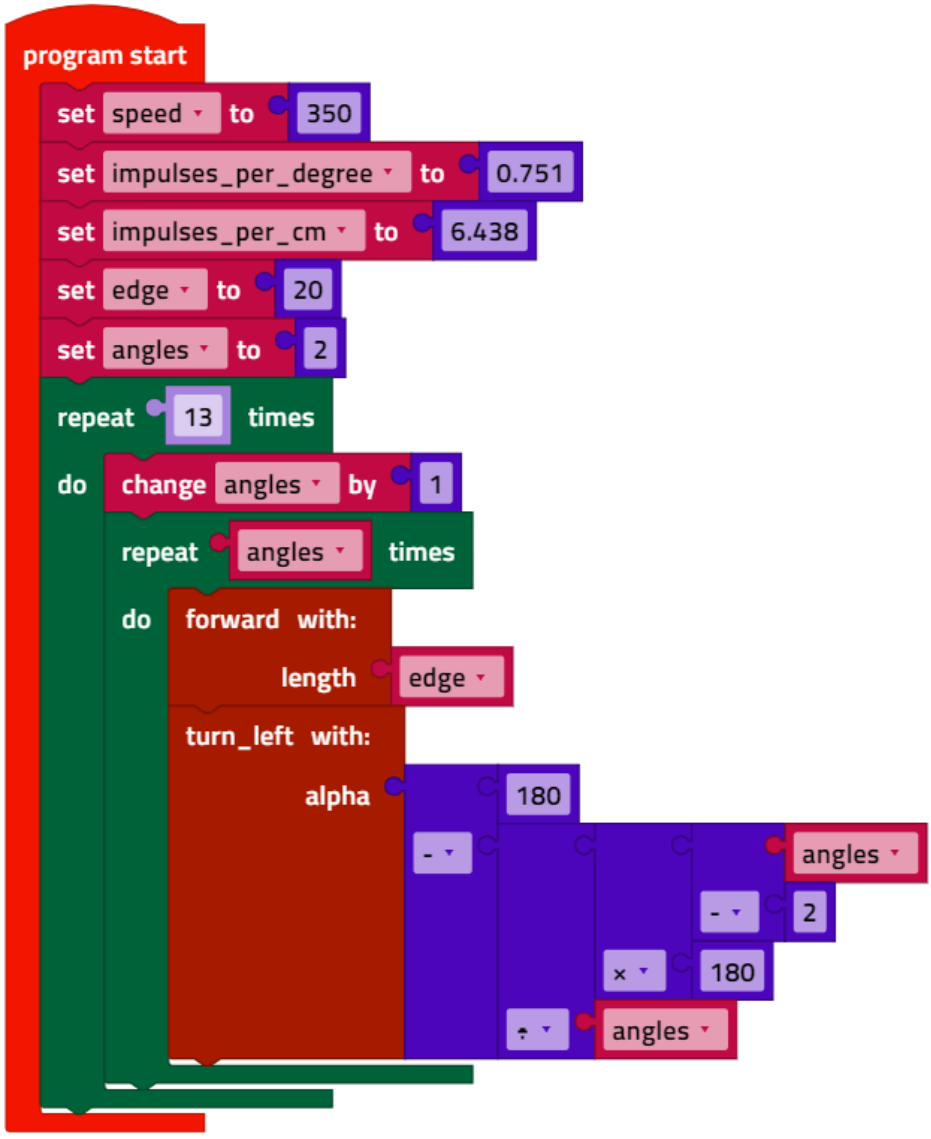


Buggy_House_of_Santa_Claus.ft

5. N- kenarlı çokgen:

Herhangi sayıda kenarı olan bir çokgenin iç açılarının toplamı (n-1)·180°'dir. Bu nedenle, herhangi sayıda kenarı olan bir çokgen çizmek için, buggy'nin dönmesi gerekir $180° - \frac{(n-1)\cdot 180°}{n}$ her taraftan sonra. Örnek çözüm, herhangi sayıda kenarı olan bir şekil çizme görevini genelleştirir ve üçgeni, her durumda kenar uzunluğu 20 cm olan 15 kenarlı bir çokgene dönüştürür.

Program (örnek):



Buggy_Drawing_Polygons.ft

Deneysel görevler (ÇÖZÜM)

5. Hedefe doğru sürün:

1a. (x,y) noktasına ulaşmak için, araba, x ve y kenar uzunluklarına sahip bir dik üçgenin hipotenüsü boyunca hareket etmelidir. Hipotenüsün uzunluğu Pisagor teoremi kullanılarak hesaplanır:

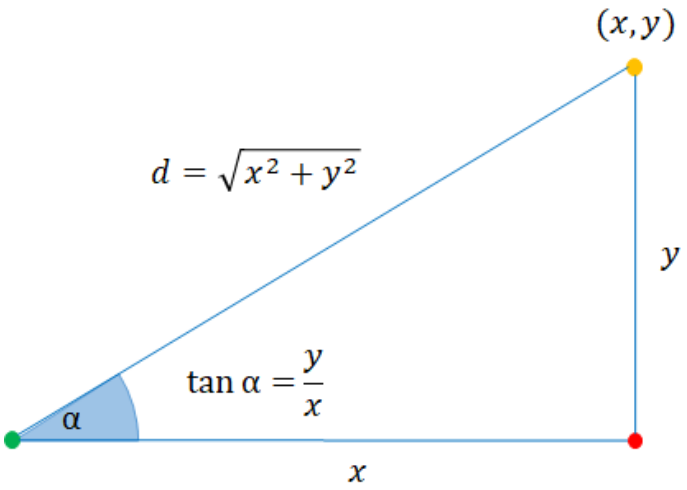
$$d = \sqrt{x^2 + y^2}$$

İç açı α trigonometri kullanılarak hesaplanabilir:

$$\tan \alpha = \frac{y}{x}$$

x eksenine dayalı δ dönüş açısı da bundan türetilmelidir. Bunu yapmanın en basit yolu, durumlara göre bir tanımdır:

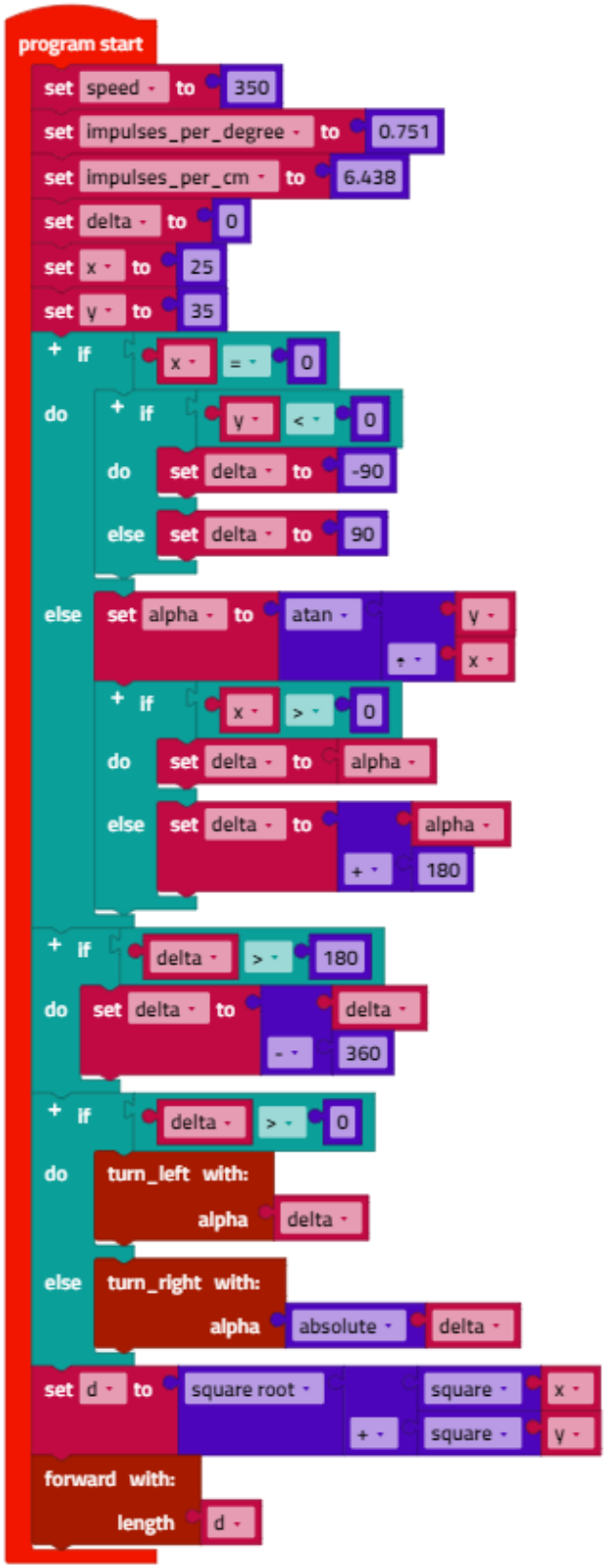
- If $x > 0$, eğer δ açısı α'ya karşılık gelir.
- If $x < 0$, eğer $\delta = 180° + \alpha$.



Matematiksel Model Koordinatları Çizimi

Son olarak, $x = 0$ ayrı olarak ele alınması gerekir, böylece bölünme olmasın 0.
eğer $y > 0$ bu durumda $\delta = 90°$, aksi halde $\delta = -90°$.

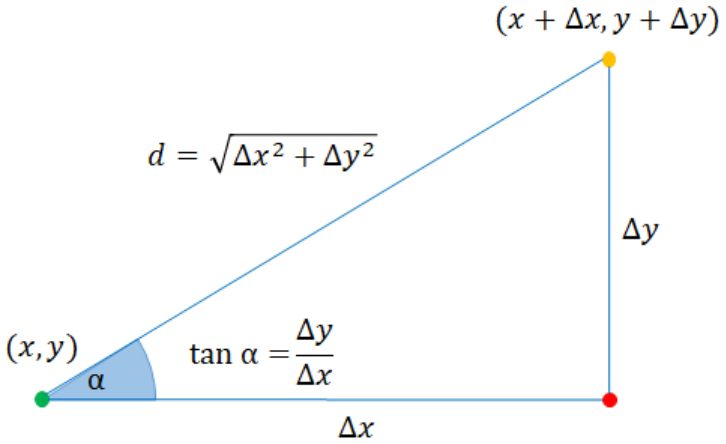
1b. Program (örnek):



Buggy_Move_2_Point.ft

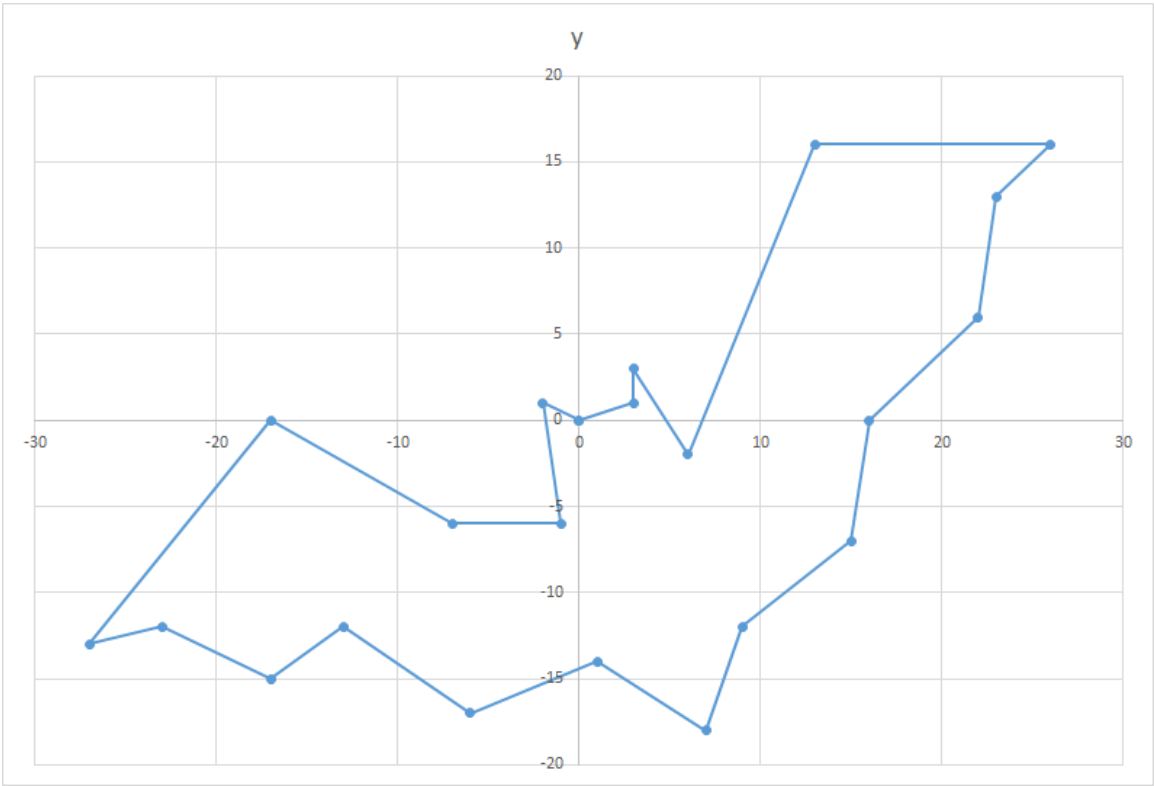
2. "Sayılarla boyama"

1a. (x,y) noktasına ulaşmak için, araba, x ve y kenar uzunluklarına sahip bir dik üçgenin hipotenüsü boyunca hareket etmelidir. Hipotenüsün uzunluğu Pisagor teoremi kullanılarak hesaplanır: $((\Delta x, \Delta y))$.



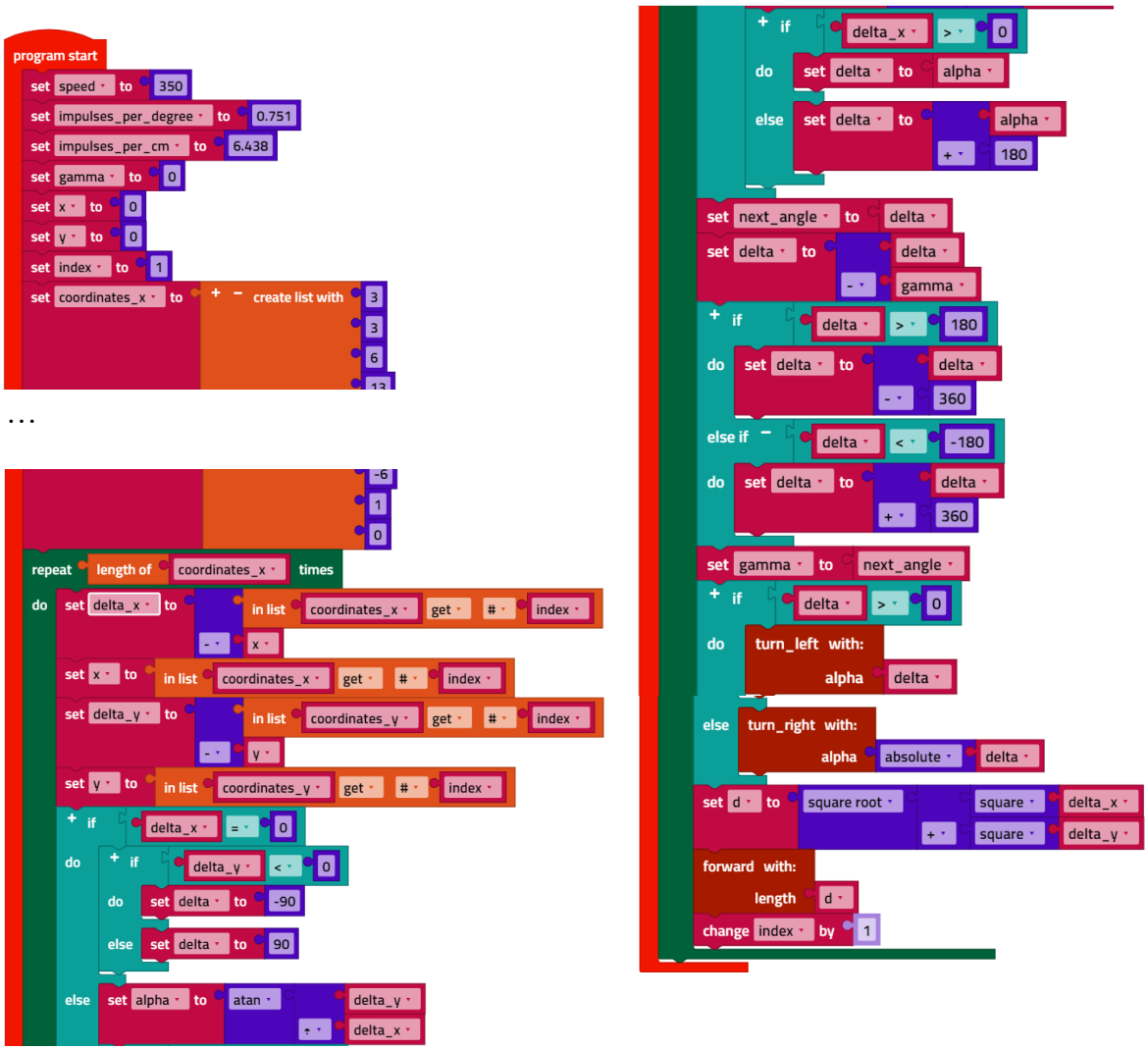
Matematiksel Model Koordinat Çizimi genelleştirilmiş

Yaklaşılacak noktalar, ilgili x ve y koordinatlarıyla iki listeye kaydedilir. Sonuç olarak aşağıdaki çizim elde edilir:



Yarasa

Program (örnek):



Buggy_Coordinates_Drawing.ft

3. Kesinlik:
Çizimlerin hassasiyetini artırmanın farklı yolları vardır:

a) Çok dar lastikler (dönüş yarıçapı sabit)
b) Daha küçük tekerlek çapı (sürülen cm başına daha fazla darbe)

c) Darbeler yerine kodlayıcının "yanlarını" saymak (motor aksı dönüşü başına iki kat daha fazla sinyal)

d) Tahrikin daha büyük indirgeme oranı (sürülen cm başına daha fazla darbe)

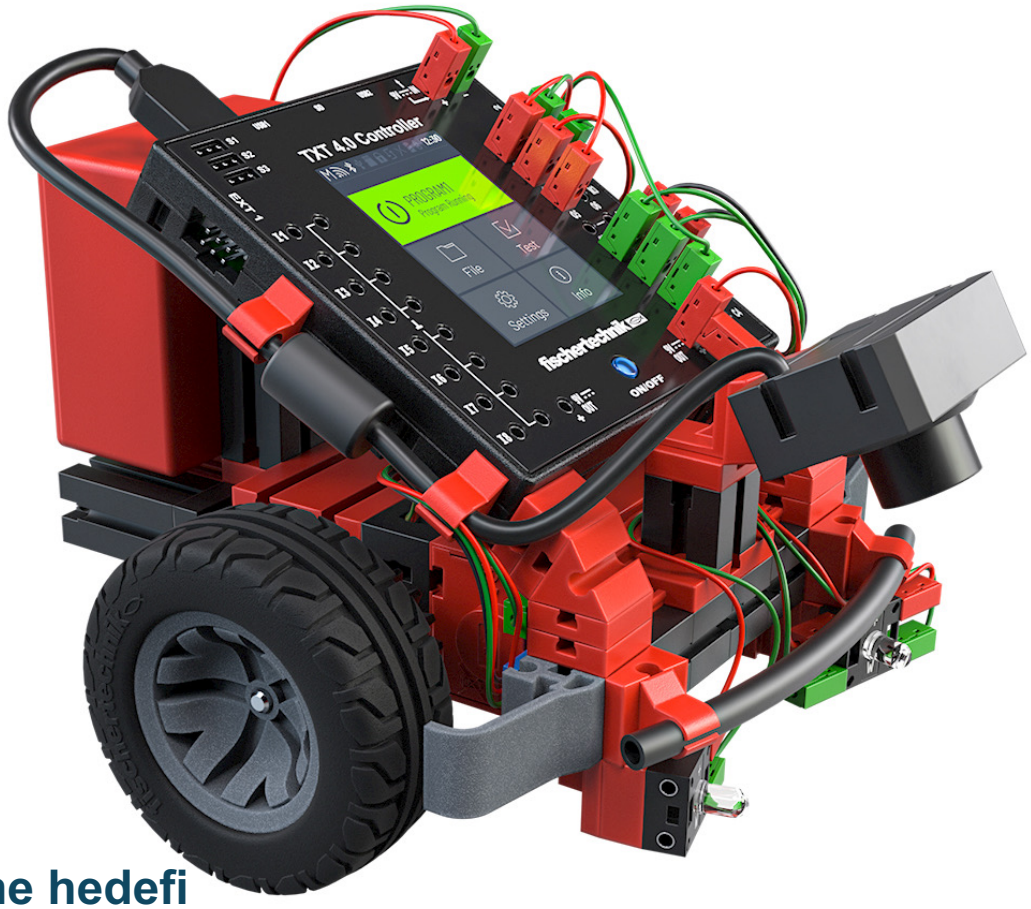
e) Daha az "yol noktası", çünkü küçük hatalar zamanla birikir.

ftrobotik

fischertechnik

BÖLÜM 07

Analog İz Takip Robotu

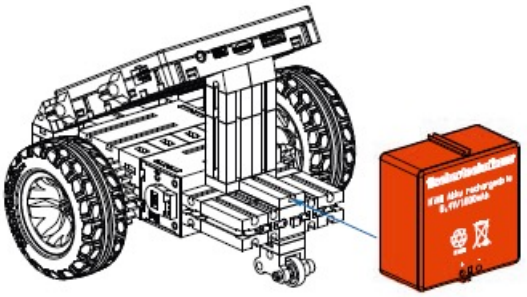


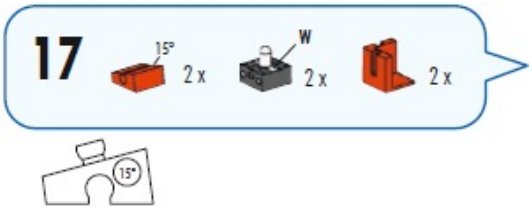
Öğrenme hedefi

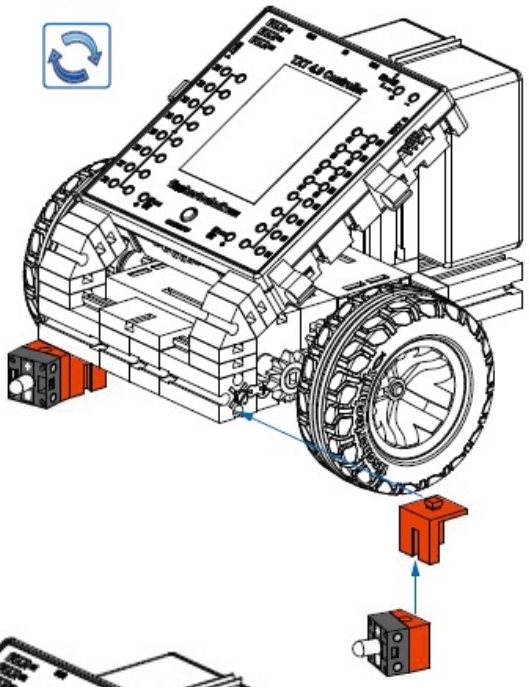
- ✓ Engel tespitinin eklenmesi (ultrason mesafe ölçümü)
- ✓ Bir kontrol cihazına renkli yüzeyler ekleme
- ✓ Görüntü değerlendirmesi, kamera kullanılarak analog kontrol
- ✓ P ve PD kontrolörleri

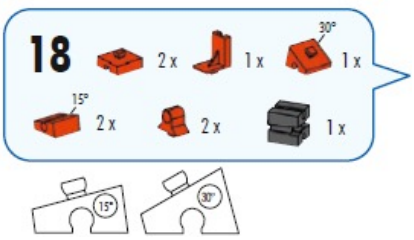
MODEL 7 Analog İz Takip Robotu

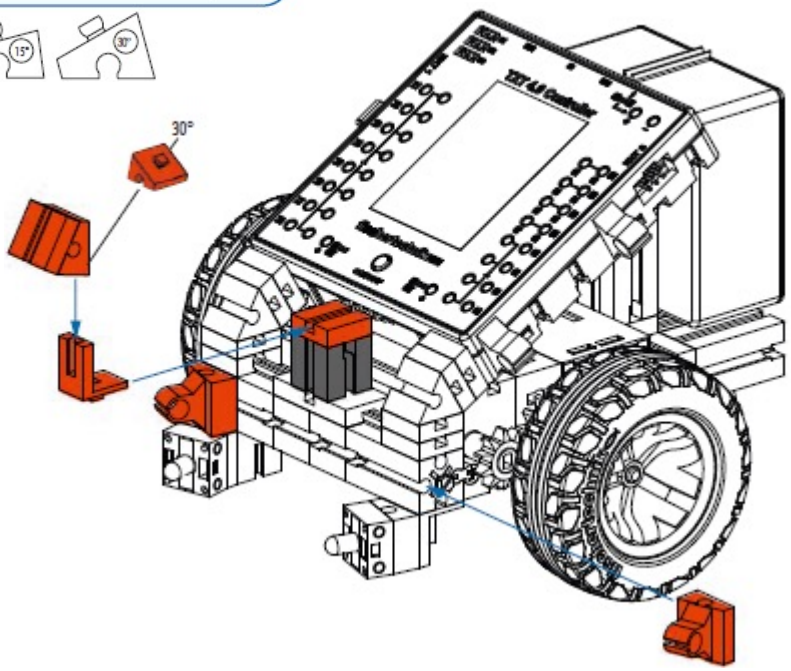
Model 8: 16. Kurulum adımı üzerinden devam edelim
Sayfa: 17

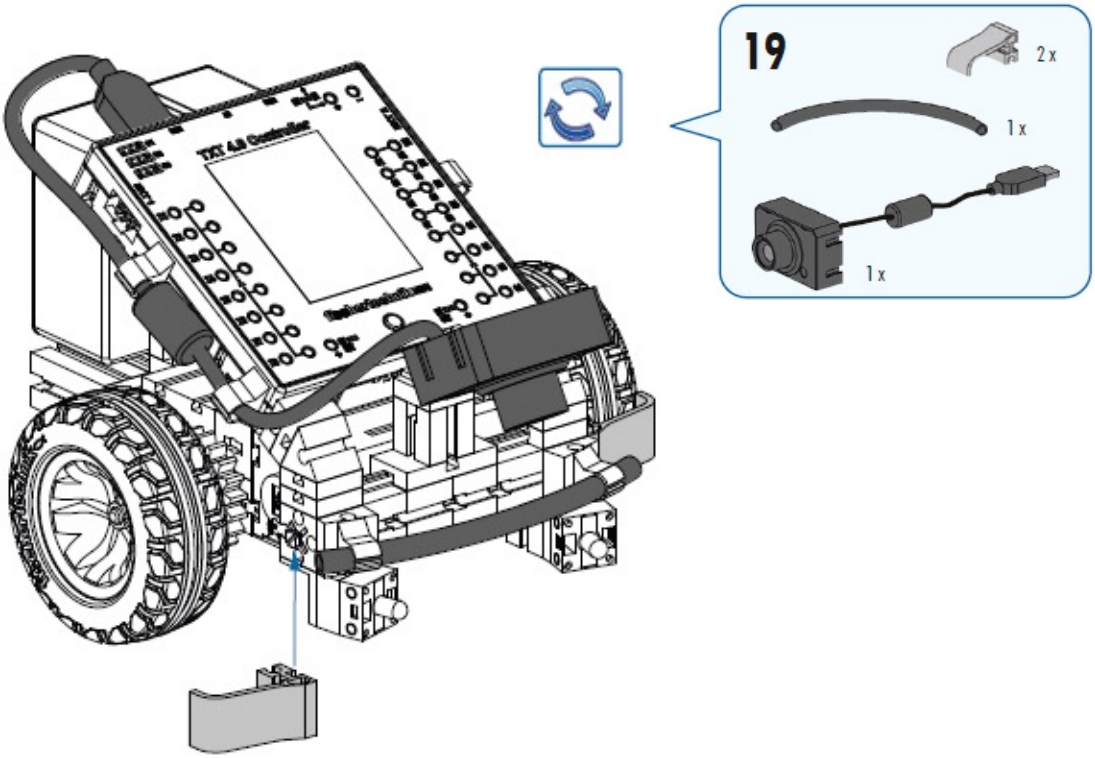


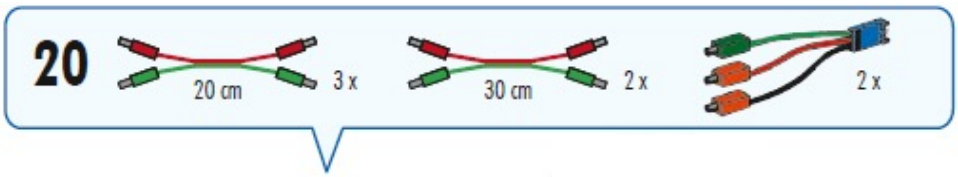


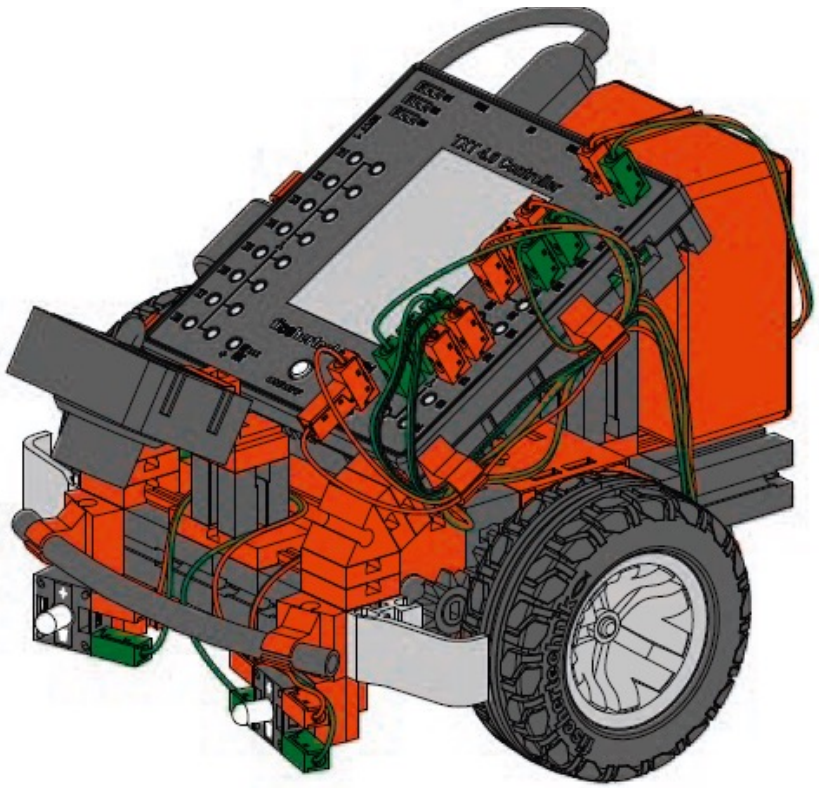




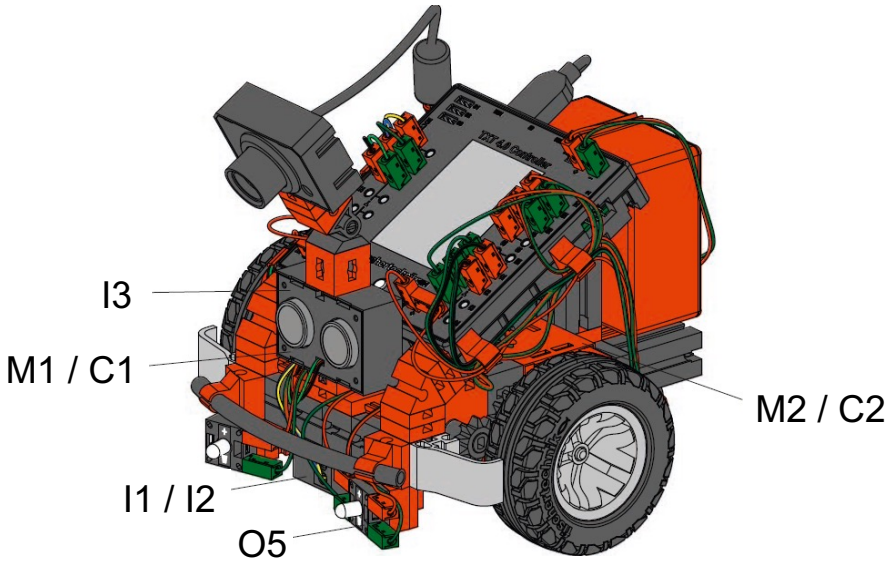



19
2 x
1 x
1 x

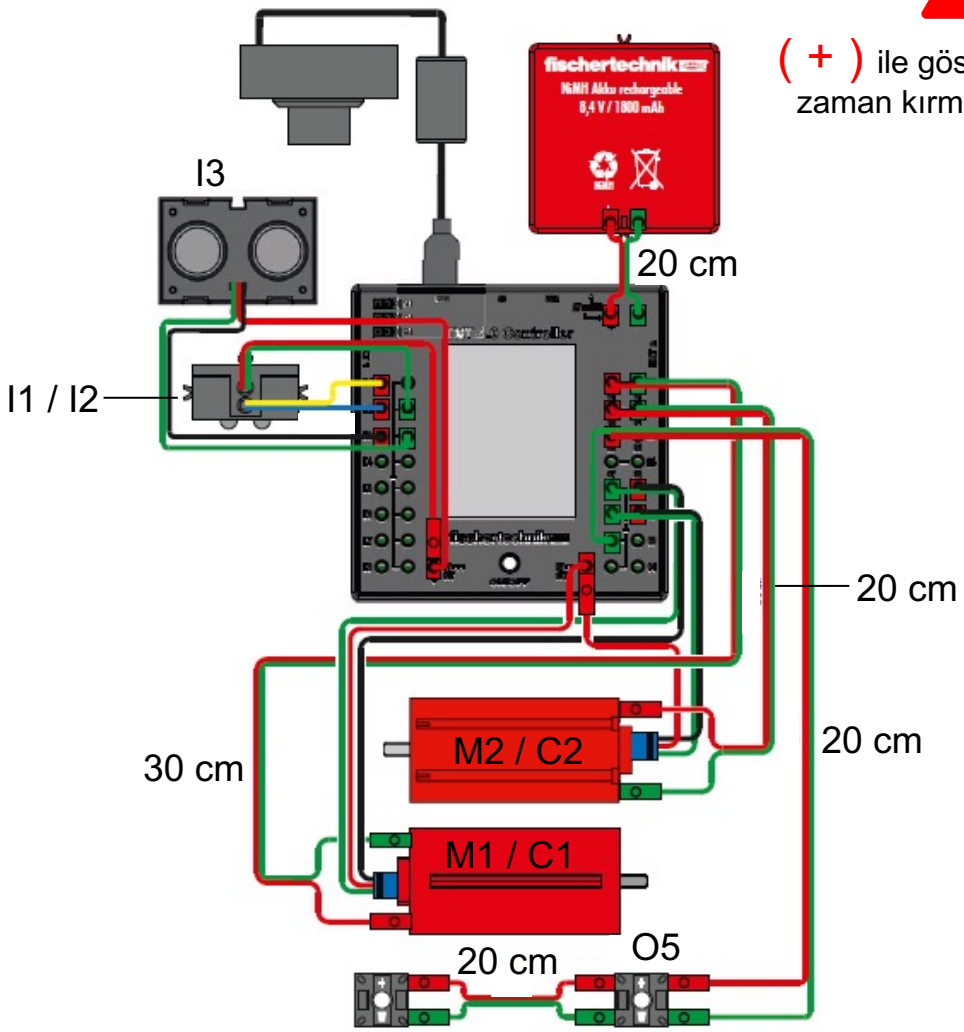

20
20 cm
3 x
30 cm
2 x
2 x



Model -7 Analog İz Takip Robotu



Devre Şeması



(+) ile gösterilen alana her zaman kırmızı soket takılır.

Analog İz Takip Robotu Öğrenme Hedefleri

Konu

İz takip cihazının (mekanik) tasarımı, görev 6'daki arabanın temel modeline dayanmaktadır. Görev sırasında, engel algılama için bir ultrason sensörü, iki ön ışık ve analog kontrol için bir kamera ile donatılacaktır.

Bir aracın (robotun) iz üzerinde dijital kontrolü (sonlu otomatlar) ve analog kontrolü.

Öğrenme hedefi

- Dijital sensörlerin işlevi
- Dijital sensörler kullanılarak basit iki nokta kontrolü
- Engel tespitinin eklenmesi (ultrason mesafe ölçümü)
- Bir kontrol cihazına renkli yüzeyler ekleme
- Görüntü değerlendirmesi olan bir kamera kullanılarak analog kontrol
- P ve PD kontrolörleri

Gerekli malzemeler

- Program geliştirme için bilgisayar veya tablet.
- Programı TXT4.0'a iletmek için USB kablosu veya BLE veya WiFi bağlantısı.
- Siyah kapalı daireli, 2 cm genişliğinde çizgili ve renkli yüzeyli ders çizelgesi.
- Engel (kutu, kitap, ...)

Daha fazla bilgi

[1] Wikipedia: *Finite automaton (state machine)*

[2] Ferdinand Wagner, Ruedi Schmuki, Thomas Wagner, Peter Wolstenholme: *Modeling Software with Finite State Machines. A Practical Approach*. Auerbach Publications,

[3] Online diagram editor for creating state diagrams (Format drawio): *https://www.diagrammeditor.de/*

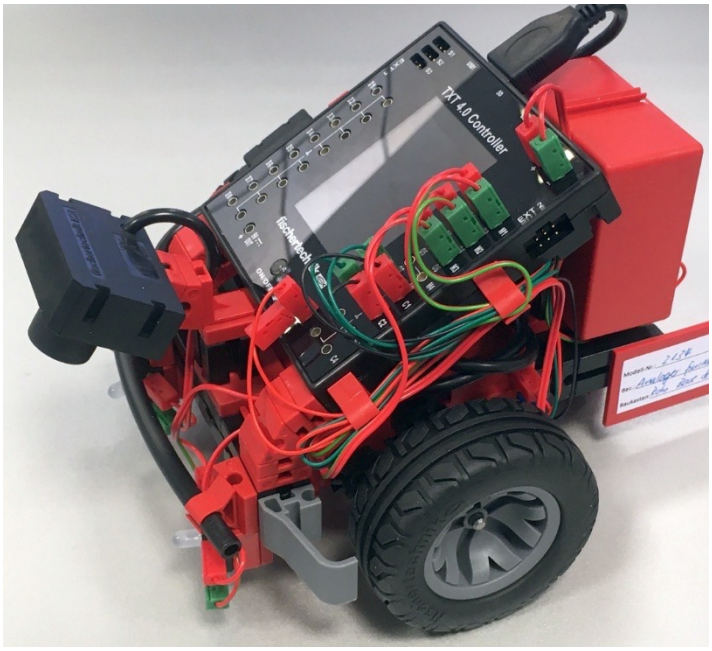
[4] Wikipedia: *Control technology*.

[5] Wikipedia: *Controller*.

[6] RN-Wissen: *Control technology*.

[7] Tim Wescott: *PID without a PhD*. Embedded Systems Programming, 10/2000, p. 86-108.

Analog İz Takip Robotu Öğrenme Hedefleri



Analog İz Takip Robotu

Programlama görevleri (GÖREV)

1. Renkli yüzeylerle kontrol:

Şimdi, arabaya kamera ve iki ön far da ekleyeceğiz. Kamera USB1 portuna bağlı ve iki LED O5 çıkışına paralel olarak bağlanıyor.
Ayrıca foto-transistörü arabanın yan tarafına takın ve I4'e bağlayın.

Parkurun sağında ve solunda birden fazla renkli yüzey göreceksiniz. Kamerayı kullanarak buggy'ye bu renkli yüzeyler üzerinden komutlar verebilirsiniz - dön, flaş yap veya ses çıkar.

4a. Buggy'nin farklı renkli yüzeylere nasıl tepki vermesi gerektiğini belirleyin.
4b. Renkli yüzeylerdeki renklerin HEX kodunu belirleyebileceğiniz kısa bir yardımcı program yazın.
4c. Ardından Blockly programınızı genişletin, böylece buggy parkuru takip ederken renkli yüzeylerin renklerini tanımlasın ve ardından bunlara göre tepki versin.

İpucu: Renk tanımayı belirlenen HEX kodlarına göre ayarlayın ve 1 toleransını seçin. Testleriniz sırasında hatalar tespit ederseniz, bu değeri düşürmeniz gerekecektir.
4d. Arabanıza bir "alacakaranlık anahtarı" ekleyin, böylece iki ön far foto-transistöre bağlı olarak otomatik olarak etkinleştirilir veya devre dışı bırakılır.

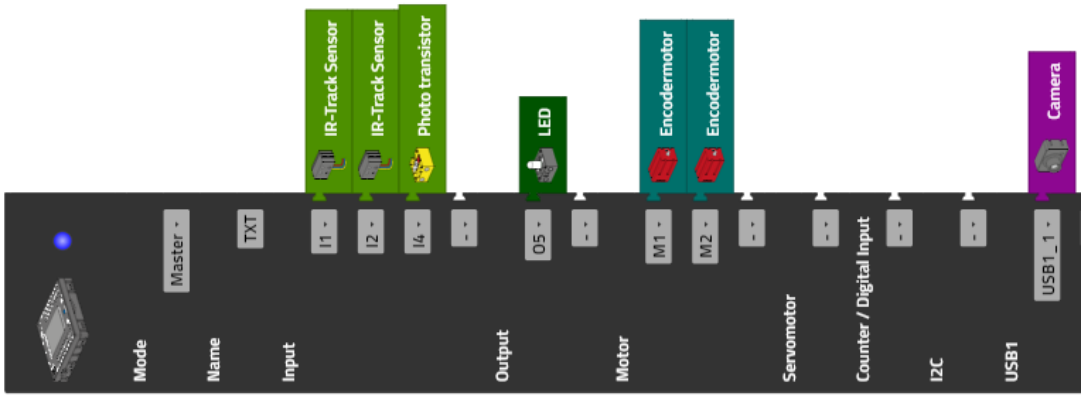
Analog İz Takip Robotu Öğrenme Hedefleri

Programlama görevleri, sonlu otomatların önemini ve kavramını anlamak için özellikle yararlıdır. Lütfen, programlama başlamadan önce durum diyagramının optimize edilmesi ve mümkün olan en az duruma indirilmesi gerektiğini unutmayın.

Deneysel görevler bir P ve bir PD kontrolörü ile ilgilidir.

Programlama görevleri (ÇÖZÜM)

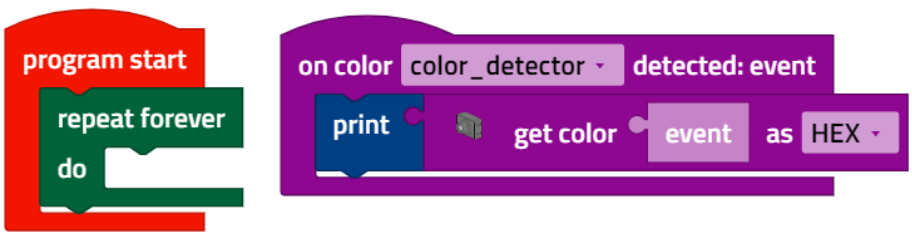
1. Engel Renkli yüzeylerle kontrol
Kamera ve ön farın yapılandırılması:



1a. Aşağıdaki örnek programda, buggy aşağıdaki renkli alanlara tepki verir:

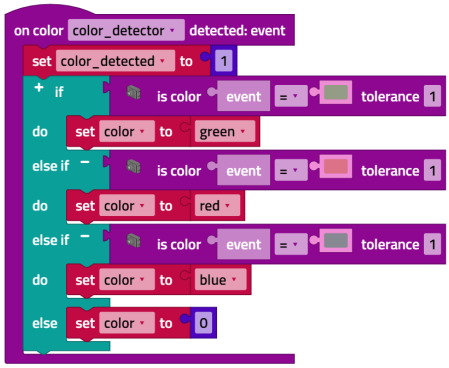
- Kırmızı: Korna
- Yeşil: Yerinde 180° dön
- Mavi: Bir kez hızlıca yanıp söner (0,1 sn)

1b. Kamera yazılımı tarafından algılanan renk tonu büyük ölçüde aydınlatmaya bağlıdır. Bu nedenle, renk bir RGB değeri (HEX) olarak seçilmelidir. Bu değeri konsolda görev 4'teki (deneysel görev 1a) "RGB_Colorcode.ft" programının basit bir çeşidini kullanarak çıktı olarak alabilirsiniz.

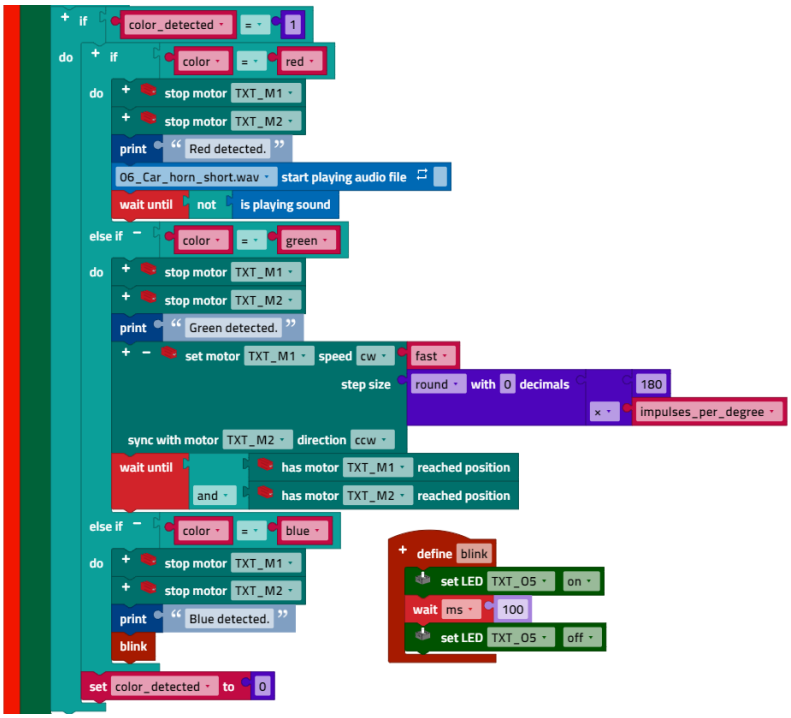


HEX_Colorcode.ft

1c. Belirlenen hex değerleri renk algılamada renk değerleri olarak girilir. Tolerans, algılamanın hassasiyetini ayarlamak için kullanılabilir.
Renk tanıma, bir değişken (“color_detected”) aracılığıyla sinyallenir.
Program bölümleri (örnek):

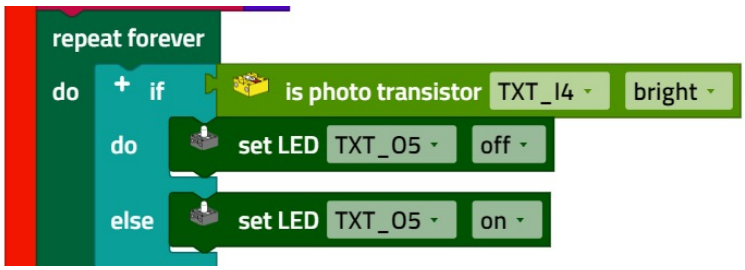


Buggy_Line_Follower_with_Color_Recognition_digital.ft



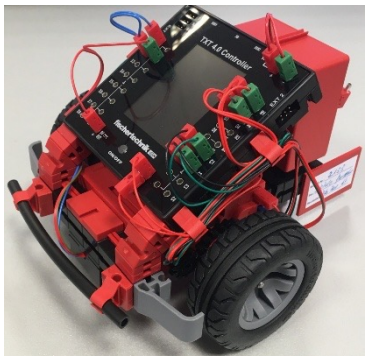
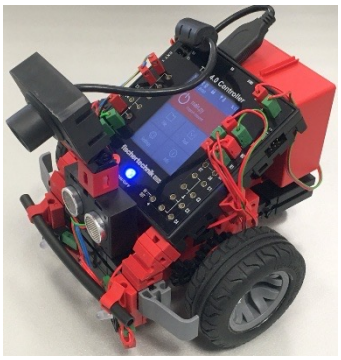
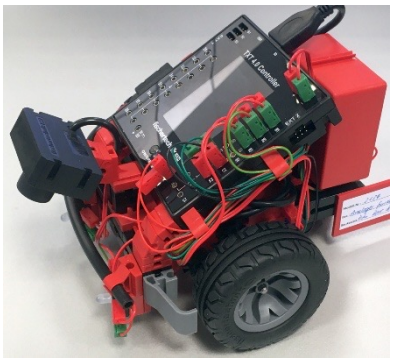
Buggy_Line_Follower_with_Color_Recognition_digital.ft

4d. Ön farın otomatik (devre dışı) bırakılmasına ilişkin program kesiti (örnek):



Buggy_Line_Follower_with_Color_Recognition_digital.ft

Deneysel Görevler

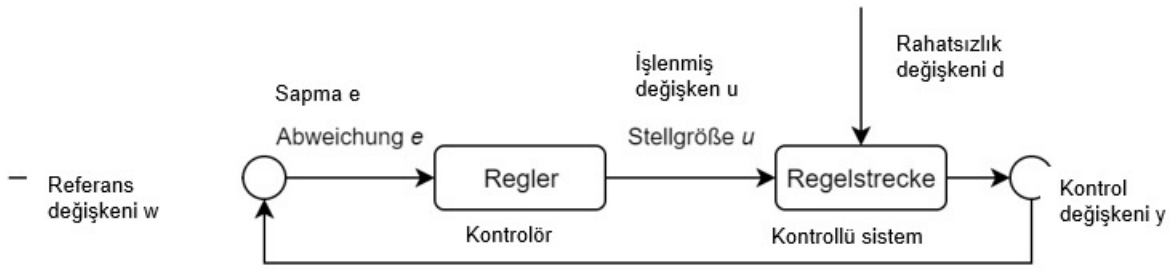
Deneysel Görevler İçin Görevler

Deneysel görevler (GÖREV)

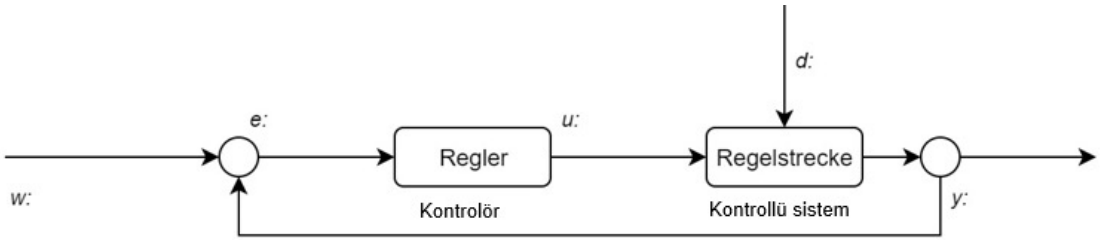
Arabayı yalnızca kamerayı kullanarak da kontrol edebiliriz. Bunu yapmak için, "analog iz takipçimizden" diğer tüm sensörleri (ultrason, iz sensörü) kaldıracağız.
İz takipçisi arabasını buna göre dönüştürün.
Artık arabayı kameranın çizgi algılama işlevine sahip bir kontrol cihazıyla donatabiliriz.

1. Kamera ile parça tespiti:
Kameranın çizgi algılama fonksiyonunu kullanarak pistin yerini belirleyebiliriz. Pistin sağında ve solunda kamera tarafından çizgi olarak algılanabilen renkli yüzeyler olduğundan, kamera tarafından algılanan pistin rengini değerlendirmemiz gerekecektir.
Birden fazla çizgiyi algılayabilen ve algılanan çizgilerin RGB renklerini konsola çıkarabilen bir Blockly programı yazın.
Şimdi, arabanızı pist boyunca farklı noktalara yerleştirin. Siyah pisti kolayca ve güvenilir bir şekilde tanımlamak için hangi özellikler kullanılabilir?

2. P kontrolörü ile takip takip cihazı:
Şimdi, buggy'nin basit bir orantılı kontrolcü kullanarak otomatik olarak yolu takip etmesini istiyoruz.
Yol takipçisini bir kontrolör olarak düşünün.



2a. İz takipçisinin hangi değişkenleri kontrol döngüsündeki *w, e, u, d* ve *y* değerlerine karşılık gelir? Aşağıdaki şekli buna göre etiketleyin:



2b. Orantılı bir kontrolör k_p) kullanarak arabayı parkurun yolu boyunca yönlendiren bir Blockly programı yazın.
Bunu yapmak için deneysel görev 1'de geliştirdiğiniz yol tespitini kullanın.

İpucu: Motorlar için maksimum hız olarak 350'yi seçin. Yönü düzeltmekten sorumlu motorun hızını (eğer pistin solundaysa: sağ motor, eğer pistin sağındaysa: sol motor) her 20 ms'de $k_p \cdot e$ ile azaltın.
Deneyinize, k_p=1 olan düz bir parça üzerinde orantı faktörünü k_p olarak ayarlayarak başlayın ve denetleyici tüm parçayı salınım yapmadan takip edene kadar değeri kısa artışlarla artırın

3. PD kontrolörlü takip cihazı:
Orantılı denetleyici, seyahat yönünü nispeten yavaş bir şekilde düzeltir; bu nedenle, dar virajlarda buggy rotayı kaybedebilir.
Denetleyici, rota sapmasındaki değişim miktarını ölçen bir "D" faktörü (diferansiyel faktör k_d) ile iyileştirilebilir: PD denetleyici, rota yönündeki ani değişikliklere (dar virajlar) daha hızlı tepki verir ve aynı şekilde aşırı atmayı azaltır.

3a. Deneysel görev 2'den orantılı denetleyiciniz için denetleyici verilerini (rotadan sapma, orantılı düzeltme değeri) konsoldaki rota boyunca bir yolculukta çıktı olarak alın, ardından bu verileri bir elektronik tablo programına kopyalayın:
Denetleyici verilerini grafiksel olarak görüntüleyin.

- Tabloya, ray sapmasındaki farkı (geçerli sapmadan son sapmayı çıkararak) hesaplayan bir sütun ekleyin.
- Başka bir sütunda, verilen diferansiyel faktör k_d 'den bir PD kontrolörünün düzeltme değerini hesaplayın. Grafikte, faktörün rota düzeltmesi üzerindeki etkisini göreceksiniz.
- PD kontrolcünüz için uygun görünen k_d değerini seçin.

3b. Pist takipçinizdeki P kontrolcüsünü genişletin, böylece bir PD kontrolcüsü olsun. Görev bölümü 3a k_d 'deki simülasyonda seçilen değerin uygun olup olmadığını kontrol etmek için test çalıştırmalarını tamamlayın. Gerekirse küçük artışlarla düzeltin.

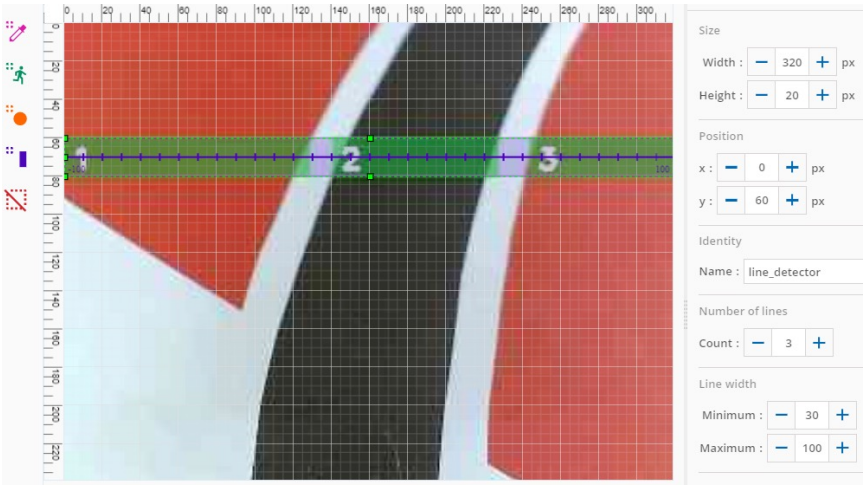
İpucu: Kontrolcüye ait değerleri konsola çıktı olarak verip elektronik tablolama programında değerlendirirseniz, kontrolcüdeki küçük "sapmaları" bile kolayca tespit edebilir ve k_d'yi ayarlarken bunları dikkate alabilirsiniz.

Deneysel Görevler

Deneysel görevler (ÇÖZÜM)

1. Kamera ile parça tespiti:

1a. Hat algılamanın yapılandırılması:

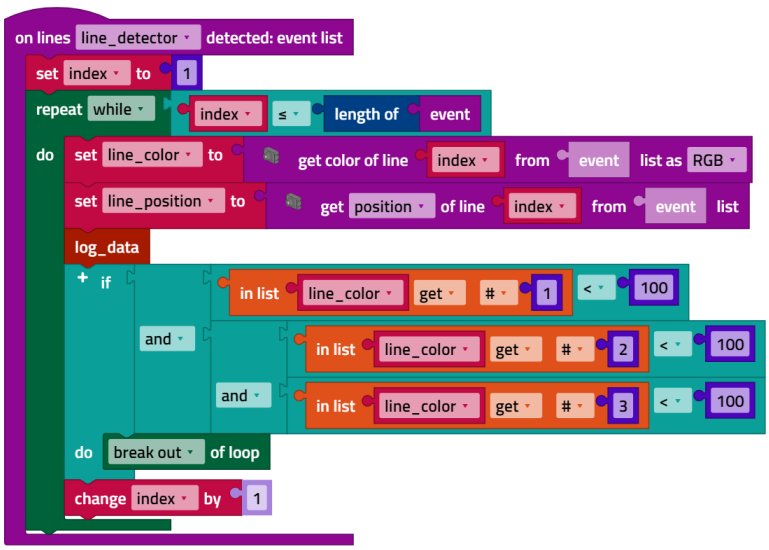


Çizgi algılama, kameranın görüş alanının tüm genişliğini kapsamalıdır. Dar bir şerit olarak seçilmelidir, çünkü algılama çizginin alanı etrafına bir kare yerleştirir; aksi takdirde çizgi bir eğride çok geniş olur.

Çizginin sağında ve solunda renkli yüzeyler olabileceğinden, çizgi algılama en az iki çizgiye ayarlanmalıdır.

Siyah çizgiyi algılamanın en güvenilir yolu renk kullanmaktır: RGB değerlerinin hepsi 100'ün oldukça altındadır, renkli yüzeyler için ise üç RGB değerinden en az biri 100'den büyüktür.

1b. Birden fazla parçanın rengini ve konumunu görüntülemek ve siyah parçayı tanımak için program özeti (örnek):



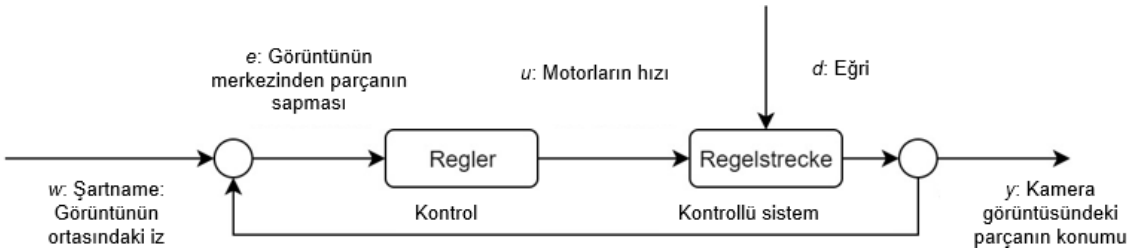
Buggy_Line_Recognition.ft

Verileri konsola çıkarmak için ayrı bir işlev ("log_data") sağlamak yararlıdır, böylece veriler zaman açısından kritik olay rutininden herhangi bir zamanda hızla silinebilir (veya hızla eklenebilir).

Buggy_Line_Recognition.ft

2. Orantılı kontrolörlü takip cihazı

2a. Kameralı (çizgi algılama) iz takip cihazı, kontrol devresi olarak gösterilmiştir:

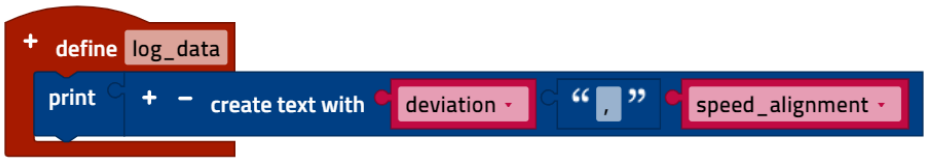


İzleyici kontrol döngüsü

2b. Orantılı denetleyicinin ayarlanması:
Kameranın çizgi algılama penceresinin konumu, denetleyicinin önizlemesini değiştirmek için kullanılabilir.

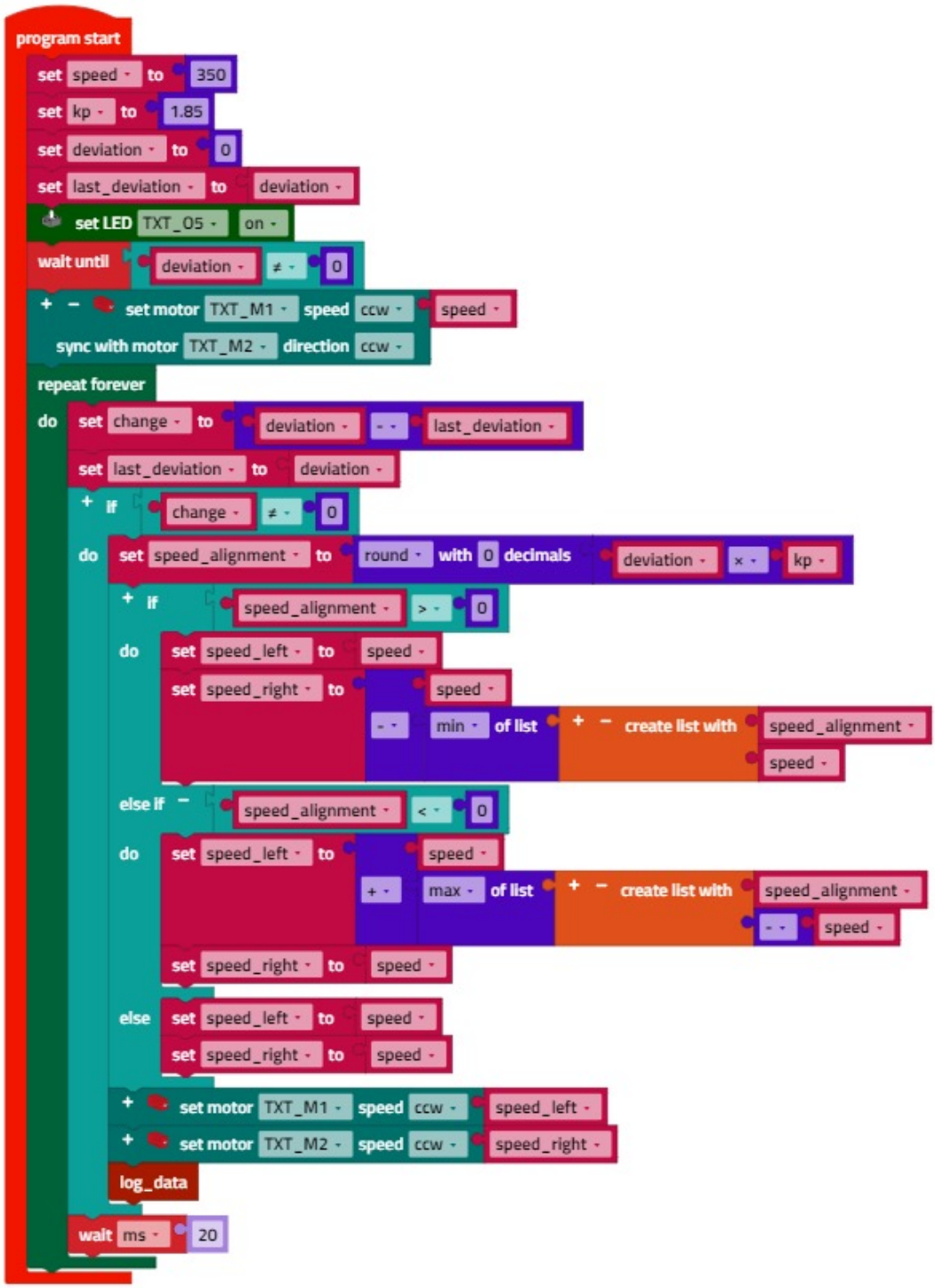
Amplifikasyon (orantılılık faktörü) K_p, düz bir yolda yapılan deneylerle belirlenir.
50/sn'lik bir kontrol frekansında (her 20 ms'de bir kontrol), uygun bir değer 1,5-2 civarında olacaktır.

Buggy daha sonra yavaşça piste yaklaşacak ve salınım yapmaya başlamayacaktır. Kontrolör değerleri konsolda "log_dat" fonksiyonu kullanılarak çıkış olarak verilebilir:



Buggy_Line_Follower_with_P_Controller.ft

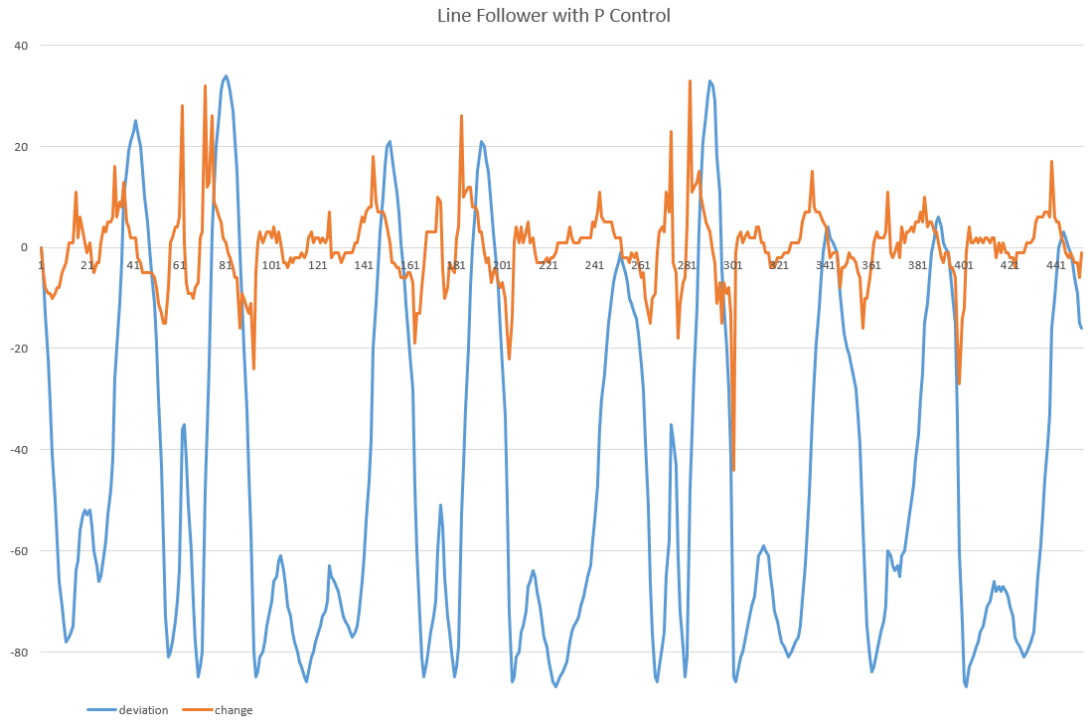
Programlama (örnek): Orantılı denetleyicili ($k_p = 1.85$) takip eden parça



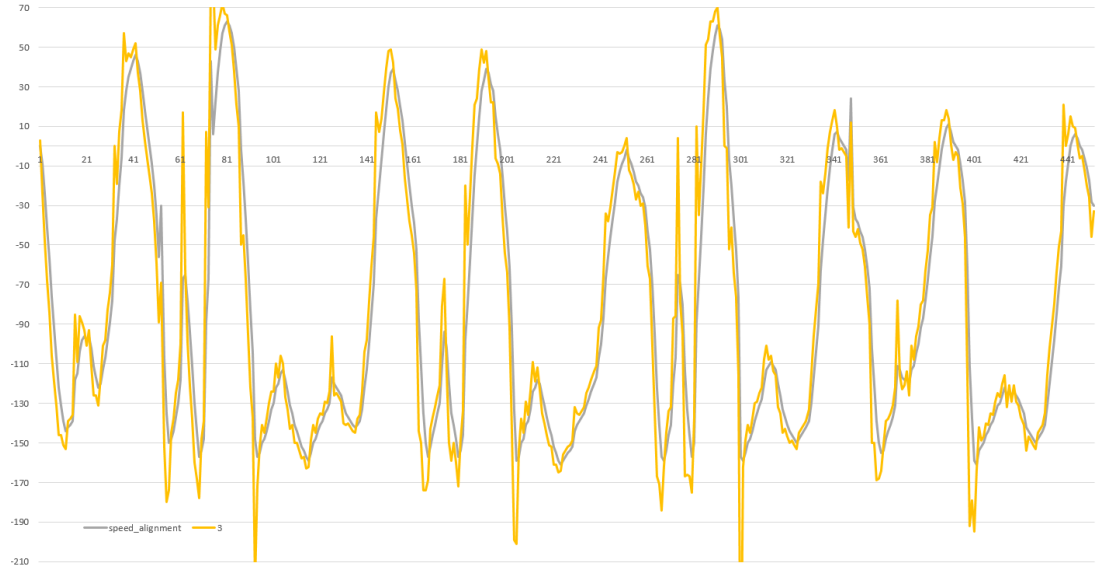
Buggy_Line_Follower_with_P_Controller.ft

3. PD kontrolörlü takip cihazı:

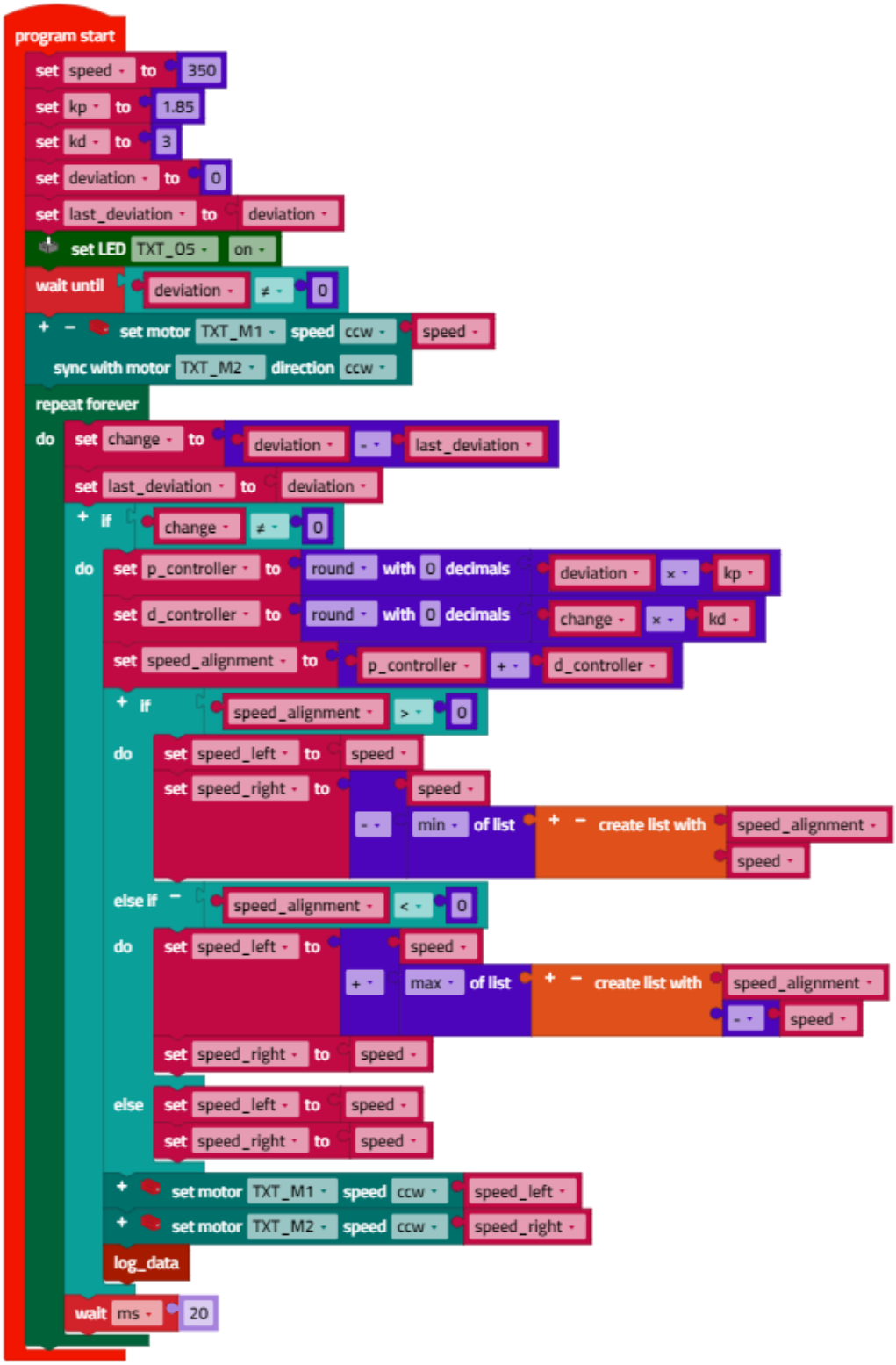
3a. P kontrolöründen gelen kontrolör verileri: İzden sapma (mavi çizgi), değişim miktarı (turuncu-kırmızı çizgi):



P kontrolörü (gri çizgi) ve PD kontrolörü için düzeltme değerleri; burada, $K_d = 3$ (sarı çizgi) ile:



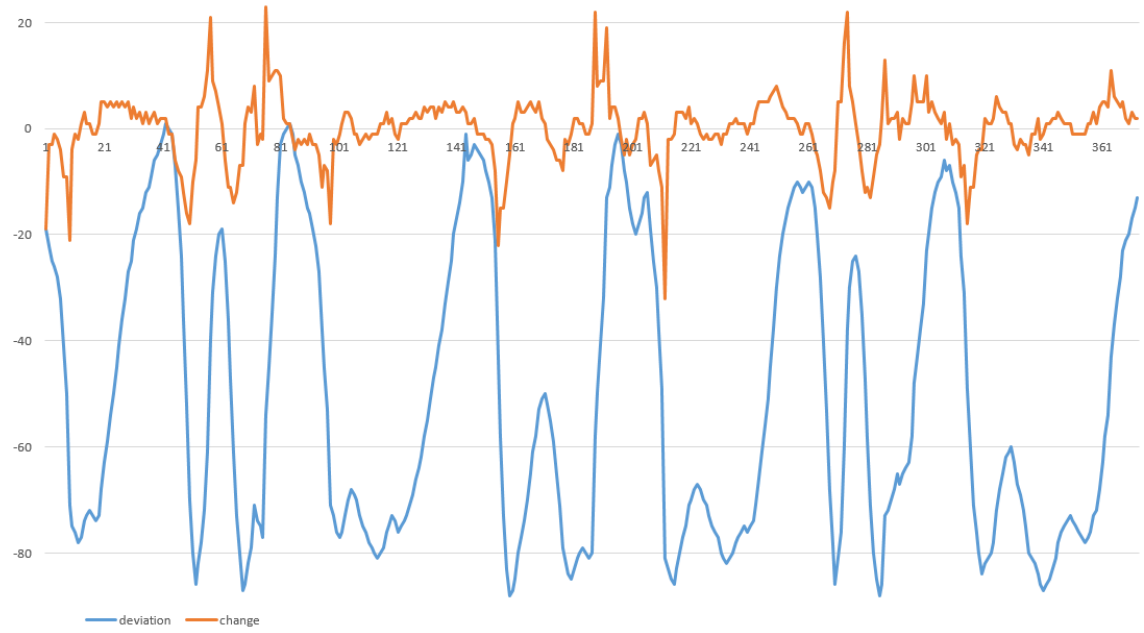
3b. Program (örnek) PD kontrolörü ile takip eden parçayı ($k_p = 1.85$ ve $k_p = 3$ ile):



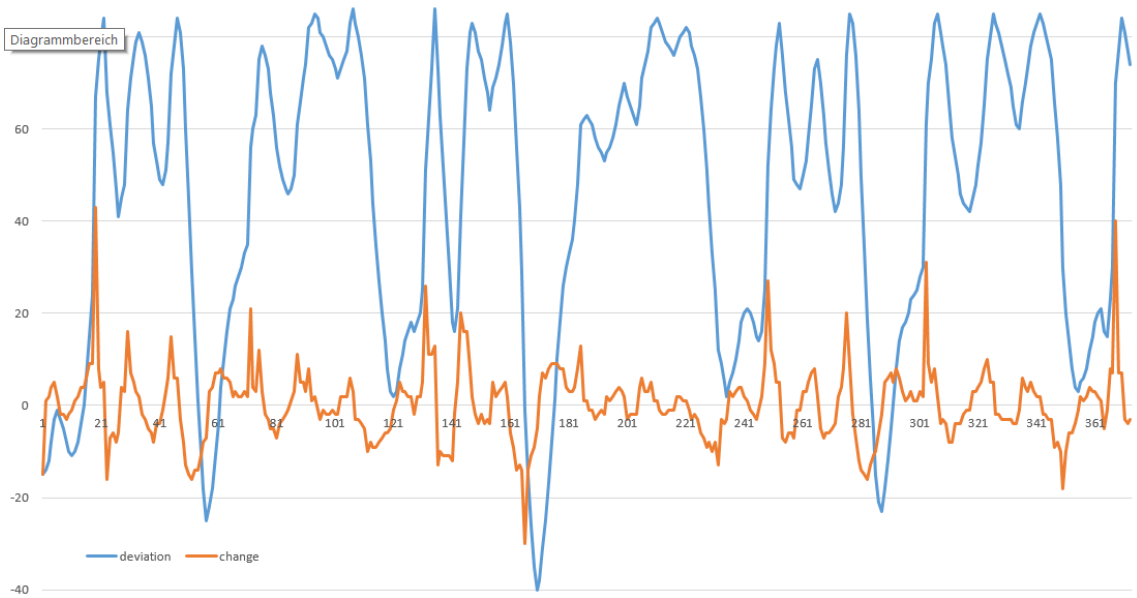
Buggy_Line_Follower_with_PD_Controller.ft

“log_data” fonksiyonu

PD kontrolöründen gelen kontrolör verileri: İzden sapma (mavi çizgi), değişim miktarı (turuncu-kırmızı çizgi



Saat yönünün tersine hareket için kontrol süreci



Saat yönünde seyahat için kontrol süreci