عنوان:

رباتیک و نحوه آشنایی با مسابقات

نویسندگان:

حمید بادامی نجات– علیرضا صالحی

کلمات کلیدی:

مسابقات رباتیک ، رباتیک ، میکرو موس، مسیر یاب، الگوریتم

چکیده:

در این این مقاله شما با مسابقات رباتیک آشنا شده و همچنین راجع به الگوریم های ربات های میکرو موس و مسیر یاب به صورت آشنایی توضیح داده شده است. در ضمن قوانین مسابقات نیز در فایلی که توسط دانشگاه آزاد تبریز تهیه شده گنجانده شده است.

رو باتیک

روبات یک ماشین هوشمند است که قادر است در شرایط خاصی که در آن قرار می گیرد کار تعریف شده ای را انجام دهد و همچنین قابلیت تصمیم گیری در شرایط مختلف را نیز ممکن است داشته باشد. با این تعریف می توان گفت روبات ها برای کارهای مختلفی می توانند تعریف و ساخته شوند. مانند کارهایی که انجام آن برای انسان غیر ممکن است.

علم روباتیک را می توان به سه شاخه اصلی تقسیم کرد:

- الکترونیک : که شامل سنسورها (حسگرها) و مدارات الکترونیکی می باشد.
- مکانیک : که شامل محرک ها، چرخ ها، و چرخ دنده ها و... می باشد
- کامپیوتر : وظیفه برنامه نویسی را در یک روبات ایفا می کند.

قوانین سه گانه روباتیک:

۱. یک روبات نباید به هستی انسان آسیب برساند یا به واسطه بی تحرکی زندگی انسان را به مخاطره بیندازد.

۲. یک روبات باید از قوانین و دستورات انسان پیروی کند (به غیر از مواردی که با قانون اول تضاد داشته باشد).

۳. یک روبات باید تا جایی که قانون اول و دوم اجازه می دهند از خود محافظت کند.

روبات ها انواع مختلفی دارند که هر کدام برای انجام کار خاصی توسط انسان ساخته می شوند که می توان به روبات های صنعتی، روبات های نظامی و روبات های مسابقه ای (روبات هایی که جنبه ی مسابقه ای دارند) و... اشاره کرد.

ما وارد بحث روبات های صنعتی و نظامی نمی شویم و به توضیح درباره روبات های مسابقه ای اکتفا می کنیم.

روبات های مسابقه ای :

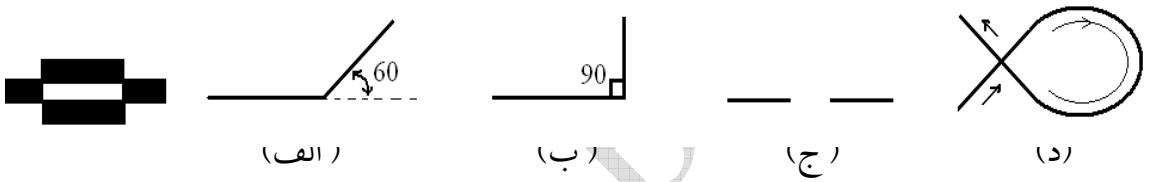
این روبات ها، روبات هایی هستند که بر اساس قوانین تعیین شده توسط مسئولین برگزاری مسابقات باید کار خاصی را انجام دهند مانند روبات تعقیب خط، لابیرنت، میکروماوس، مین یاب، جنگجو و.... که به صورت ساده توضیحاتی در مورد آنها ارائه می شود.

روبات تعقیب خط

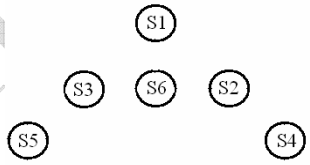
این روبات ها باید یک خط تیره را در زمین روشن و یا یک خط روشن را در زمین تیره تعقیب کنند. برای تشخیص رنگ تیره از روشن می توان از مادون قرمز استفاده کرد. زیرا همانطور که می دانید رنگ های تیره نور را جذب می کنند و رنگ های روشن نور را دفع می کنند پس با استفاده از مداراتی که در قسمت مادون قرمز به آنها اشاره شد(مقایسه کننده یاPLL) می توان این کار را انجام داد.

در این نوع روبات، نحوه ی چیدمان سنسورها بسیار مهم است یعنی ما باید با توجه به چیدمان سنسورها، الگوریتم و برنامه روبات را بنویسیم.

این چیدمان تابع قوانین مسابقات و سلیقه های افراد می باشد. مثلاً ما الگوریتم روباتی را می نویسیم که در آن مسیرهای زیر وجود دارد.



برای نوشتن الگوریتم، ابتدا می بایست سنسورها را چیده و آنها را شماره گذاری کنیم.



حال برنامه نویس می بایست طبق چیدمان سنسورها برای حالت های تعیین شده الگوریتم خود را طراحی کند.

اکنون این الگوریتم ها را برای شکل های (الف) تا (د) طبق چیدمان سنسورهای بالا پیاده سازی می کنیم.

شکل (الف):

- به طرف جلو حرکت کن ⟹ if : S1=S6=1 , S2=S3=S4=S5=0
- به طرف چپ حرکت کن ⟹ if : S1=S2=S4=S5=S6=0 , S3 =1

شکل (ب):

- به طرف چپ حرکت کن ⟹ if : S3=S6=1 , S1=S2=S4=S5=0
- به طرف راست حرکت کن ⟹ if : S6=S2=1 , S1=S3=S4=S5=0

شکل (ج):

- به طرف جلو حرکت کن ⟹ if : S1=1 , S2=S3=S4=S5=S6=0
- به طرف جلو حرکت کن ⟹ if : S6=1 , S1=S2=S3=S4=S5=0

شکل (د):

- به طرف جلو حرکت کن ⟹ if : S1=S6=S2=S3=1 , S5=S4=0
- به طرف راست حرکت کن ⟹ if : S2=1 , S1=S3=S4=S5=S6=0

این الگوریتم بسیار ساده می باشد و فقط چگونگی طراحی الگوریتم را برای خوانندگان بیان می کند. این الگوریتم در عین سادگی عیب های زیادی دارد که شما می توانید آنها را اصلاح کرده و به الگوریتم دلخواه خود تبدیل کنید.

یکی از عیب های این الگوریتم استفاده نکردن از سرعت های متفاوت است که باعث می شود روبات روان حرکت نکند.

اگر از موتورهای DC برای این روبات استفاده می کنید با استفاده از مدولاسیون عرض پالس (PWM) می توانید سرعت موتورها را کم و زیاد کنید که در مقاله قبلی راجع به کنترل موتورهای DC به طور کامل بحث شده است.

برای تشخیص خط های سیاه نیز از مدارات مادون قرمز که در مورد هر کدام از این مدارات توضیحاتی ارائه شده است ،می توانید استفاده کنید.

روبات میکروماوس:

این روبات باید در یک زمین که ۱۶×۱۶ (طول × عرض) می باشد از یکی از گوشه ها شروع بحرکت کرده و به یکی از چهار خانه وسط برسد.

- ✓ هدف این روبات یافتن مرکز زمین در کوتاه ترین زمان است.
- ✓ به زمینی که این گونه روبات ها در آن به مسابقه می پردازند Maze می گویند.

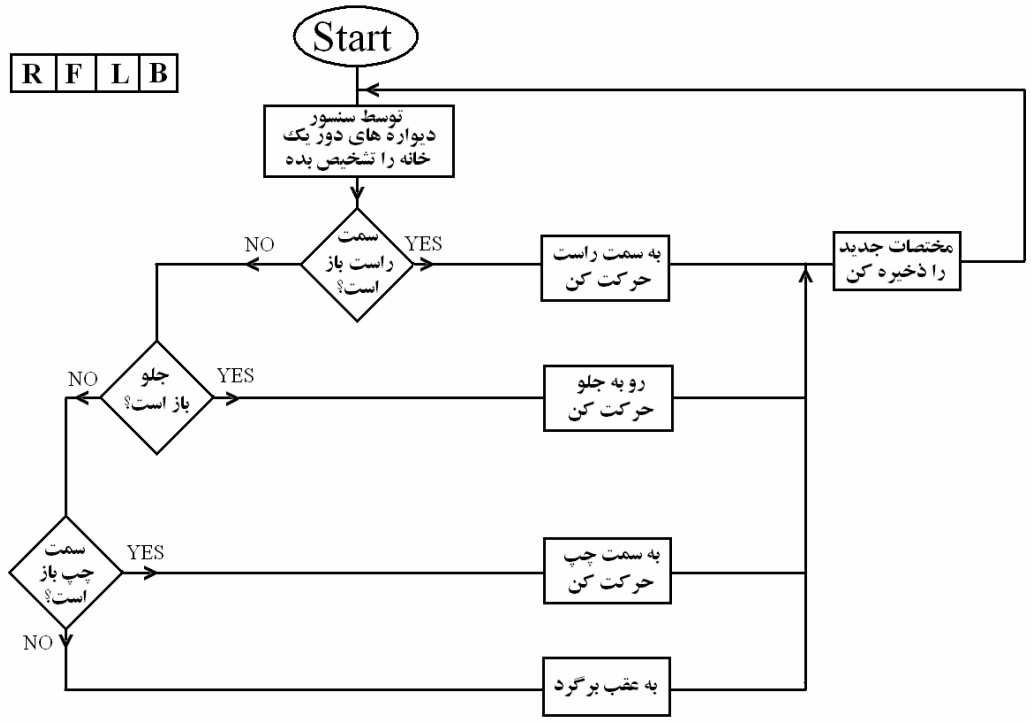
مشخصات زمین:

- این زمین دارای ۱۶×۱۶ واحد که ابعاد هر واحد ۱۸×۱۸ سانتی متر است، می باشد.
- ارتفاع دیواره ها ۵ سانتی متر و ضخامت آنها ۱۶ میلی متر است.
- کناره دیواره ها سفید و بالای دیواره ها قرمز رنگ می باشد.
- کف زمین به رنگ سیاه است.

برای رسیدن به وسط زمین روش های مختلفی (الگوریتم) وجود دارد که ما در این کتاب روش راست گرد را شرح می دهیم.

در الگوریتم راستگرد، اولویت حرکت به این صورت است که روبات ابتدا به سمت راست و بعد به سمت جلو، سپس سمت چپ و بعد به عقب باز می گردد.

یعنی روبات طبق فلوچارت زیر عمل می کند.



در این روبات زمان رسیدن به هدف بسیار مهم است پس ربات نباید خانه هایی را که بن بست هستند یا به هدف نمی رسند را بیشتر از یکبار برود. برای این کار از سیستم زیر استفاده می کنیم.

خانه هایی که نرفته =0

خانه های درست =1

خانه های بن بست =-1

دو راهی =2

سه راهی =3

برای مثال اگر بخواهیم در یک زمین ۸×۸ که مانند شکل زیر است، این الگوریتم را اجرا کنیم به صورت زیر عمل می کنیم.(خطوط پر رنگ دیوار هستند)

Start (0,0)	(0,1)	(0,2)	(0,3)	(0,4)	(0,5)	(0,6)	(0,7)
(1,0)	(1,1)	(1,2)	(1,3)	(1,4)	(1,5)	(1,6)	(1,7)
(2,0)	(2,1)	(2,2)	(2,3)	(2,4)	(2,5)	(2,6)	(2,7)
(3,0)	(3,1)	(3,2)	(3,3)	(3,4)	(3,5)	(3,6)	(3,7)
(4,0)	(4,1)	(4,2)	(4,3)	(4,4)	(4,5)	(4,6)	(4,7)
(5,0)	(5,1)	(5,2)	(5,3)	(5,4)	(5,5)	(5,6)	(5,7)
(6,0)	(6,1)	(6,2)	(6,3)	(6,4)	(6,5)	(6,6)	(6,7)
(7,0)	(7,1)	(7,2)	(7,3)	(7,4)	(7,5)	(7,6)	END (7,7)

ابتدا مختصات همه خانه ها را ذخیره می کنیم خانه مبدا (0,0) و خانه مقصد (7,7) می باشد. روبات را در خانه(0,0) قرار داده و روبات این خانه را برابر یک قرار می دهد و اولویت ها را طبق اطلاعات دریافتی از سنسورها انتخاب میکند یعنی به سمت جلو می رود(زیرا سمت راست بسته است.) با حرکت به جلو به **x** ها یکی اضافه می کند یعنی خانه(0,1)**.** در این خانه با توجه به اطلاعات سنسورها عدد 2 را قرار می دهد زیرا دو راه برای ادامه وجود دارد (از میان راه هایی که صفر هستند**.**لازم به ذکر است که در ابتدا همه خانه ها صفر هستند.) و سپس سمت راست را بر می گزیند و به خانه (1,1) وارد می شود که با این کار یکی به **y** ها اضافه می کند. روبات هیچکدام از اولویت ها را بجز برگشت به عقب را نمی تواند انجام دهد. با این کار عدد 1- در خانه (1,1) قرار داده می شود. بعد از وارد شدن دوباره به خانه ی (0,1) آنرا برابر یک قرار می دهد. سپس وارد خانه (0,2) شده و عدد 3 را در آن قرار می دهد. با توجه به اولویت ها به خانه (1,2) می رود و عدد 1 را در آن قرار می دهد. بعد به خانه(2,2) رفته و عدد 3 را در آن قرار می دهد و سپس به خانه (2,1) می رود. این خانه بن بست می باشد در این مواقع ما می گوییم تا آخرین دو راهی یا سه راهی برگرد و خانه ها را برابر 1- قرار بده پس خانه های (1,0) و (2,0) و (2,1) برابر 1- می شوند و پس از رسیدن به خانه (2,2) عدد آن را برابر 2 قرار می دهد یعنی یکی از راه ها اشتباه بود و بعد به خانه (2,3) رفته و آن را برابر 1 قرار می دهد. سپس به خانه (4,2) رفته و عدد 2 را قرار می دهد و به سمت راست حرکت کرده و پس از طی مراحلی می بیند این راه هم به بن بست می خورد پس تا خانه (4,2) برگشته

و خانه های رفته را برابر 1- کرده و خانه (4,2) را برابر یک می کند. سپس از طریق خانه (4,3) مسیر بعدی را امتحان می کند و پس از طی این مسیر می بیند این مسیر نیز اشتباه است و تا خانه (4,2) برگشته و علاوه بر اینکه اعداد خانه های این مسیر را برابر 1- قرار می دهد عدد خانه (4,2) را نیز برابر 1- قرار داده و از طریق خانه (3,2) که عدد این خانه نیز به1- تبدیل می شود به خانه (2,2) رفته و عدد آن را 1 قرار می دهد سپس به ترتیب زیر عمل می کند.

(2,3)=1 ⟶ (2,4)=2 ⟶ (2,5)=3 ⟶ (3,5)=1 ⟶(4,5)=3
(4,4)=2 ⟶ (3,4)=1 ⟶ (3,3)=-1 ⟶ (3,4)=-1⟶ (4,4)=1 ⟶ (5,4)=1
(6,5)=3 ⟶ (6,4)=2 ⟶ (5,4)=-1 ⟶ (6,4)=1 ⟶(6,3)=1
(7,3)=1 ⟶ (7,4)=1 ⟶ (7,5)= 1 ⟶ (7,6)= 1 ⟶ (7,7)=1

نحوه ی عملکرد این الگوریتم به این صورت است که پس از رسیدن به مقصد دوباره از روی خانه هایی که روی آنها اعداد 1 یا 2 یا 3 می باشد به مبدا برگردد و برای بدست آوردن رکورد بهتر دوباره مسیر مبدا تا مقصد را می پیماید. روبات می تواند در زمان داده شده (که معمولاً ۱۰ دقیقه است) چندین بار مسیر مبدا تا مقصد را بپیماید که بهترین رکورد برای روبات ثبت خواهد شد. بار اول روبات با زمین بیگانه است ولی دفعات بعدی روبات مسیرهای درست را از نادرست تشخیص داده و خانه های درست را در داخل حافظه eeprom خود ذخیره می کند تا در دفعات بعدی در زمان کمتری به مقصد برسد.

این الگوریتم، یک الگوریتم بسیار ساده می باشد که عیب آن در این است که وقتی روبات دفعه اول می خواهد مسیر را بپیماید زمان زیادی هدر می دهد. حال شما می توانید الگوریتم های بهتری را برای رسیدن به مقصد انتخاب کنید.

موتور این روبات ها بهتر است موتور پله ای باشد زیرا باید حرکات دقیقی داشته باشند. و برای تشخیص مانع می توان از آی سی LM567 (که مدار آن در قسمت مادون قرمز آمده است) استفاده کرد.

عملکرد موفق ربات:

اگر رباتی قبل از مسابقات تست وبهینه سازی شده باشد، در صورت رخ ندادن اتفاقات غیر منتظره موفق خواهد بود. این اتفاقات **۹۹**٪ عوامل شکست رباتها در مسابقات است. از جمله:

- دشارژ شدن باتری های مورد استفاده
- معیوب بودن باتری
- نقص سنسورها و حساس بودن بعضی از سنسورها به محیط اطراف
- سوختن درایور موتورها
- قطع سیم ها
- رخ دادن ایرادات نرم افزاری بر اثر نویز
- جدا شدن اجزای مختلف ربات
- و....

در تمامی مسابقات رباتیک در هر سطحی که باشندتعدادی از رباتها به دلیل نقص فنی مکانیکی، الکترونیکی و نرم افزاری از دور خارج می شوند و سایر رباتها موفقیت را بدست می آورند.