به نام خدا

پروژه:

ناوبری ربات با استفاده از GPS

نام استاد:

دکتر شیری

نام اعضای گروه:

آمنه شیخ جعفری،شقایق غرقابی

دانشگاه صنعتی امیرکبیر– گروه مستقل رباتیک

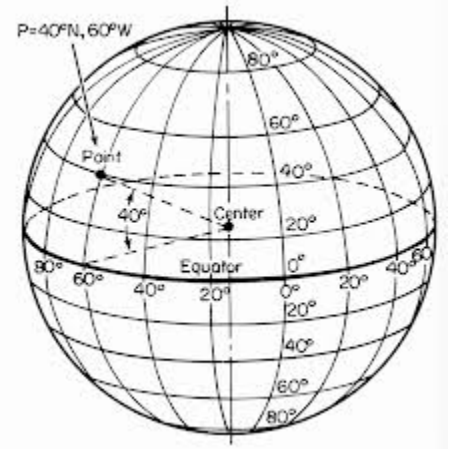
نیمسال اول تحصیلی ۹۲-۹۱

فهرست مطالب

۱ سنسورها

GPS ۱-۱

توضیحات لازم راجع چند اصطلاح



۱-۱-۱ عرض جغرافیایی

زمانیکه به یک نقشه نگاه می کنیم ، عرض های جغرافیایی همچون خطوط افقی کشیده شده اند. خطوط عرضی در واقع خطوط موازی هستند که دارای فاصله یکسان باهم می باشند. هر درجه از عرض جغرافیایی تقریبا ۱۱۱ کیلو متر است . برای بخاطر آوردن عرض جغرافیایی تصور کنید که آنها همانند پله های یک نردبان هستند.درجات عرض جغرافیایی از ۰ تا ۹۰ درجه شمال و جنوبی کشیده شده اند. درجه ۰ درست روی خط استوا است ، استوا خطی فرضی استکه سیاره ما را به دو نیمکره شمالی و جنوبی تقسیم می نماید. ۹۰ درجه شمالی در قطب شمال و ۹۰ درجه جنوبی منطبق بر قطب جنوب است .

۱-۱-۲ طول جغرافیایی

خطوط عمودی طول جغرافیایی نصف النهار نیز نامیده می شوند. آنها در قطب ها به هم می پیوندند و هر چه به استوا نزدیکتر می شوند پهن تر میشوند . فاصله طول های جغرافیایی از هم در روی خط استوا در حدود ۱۱۱کیلو متر میباشد. ۰ درجه طول جغرافیایی در گرینویچ انگلستان قرار دارد . ۱۸۰ درجه شرقی و ۱۸۰ درجه غربی جای است که خط بین المللی روز نامیده شده و در اقیانوس آرام واقع است. گرینویچ ، محل رصد خانه سلطنتی بریتانیا ، در سال ۱۸۸۴ توسط کنفرانس بین المللی جغرافیایی بعنوان مکان نصف النهار مبدا تعیین گردید.

۱-۱-۳ عرض و طول جغرافیایی چگونه با هم تعامل دارند ؟

برای تعیین محل دقیق یک مکان ، طول و عرض جغرافیایی به دقیقه و ثانیه تعیین می گردد. هر درجه ۶۰ دقیقه و هر دقیقه به ۶۰ ثانیه تقسیم می گردد. ثانیه ها می توانند برای دقیق تر شدن مکان ها به دهم ثانیه ، صدم و حتی هزارم ثانیه تقسیم گردند . برای مثال وقتی می گوییم شهر ارومیه در ۳۷ درجه و ۳۴ دقیقه شمالی و ۴۴ درجه و ۵۸ دقیقه شرقی واقع شده است یعنی این شهر با مختصات فوق در نیمکره شمالی و شرقی واقع گردیده است .

هر درجه ۱۱۱ کیلومتر است و هر درجه نیز ۶۰ دقیقه است با این اوصاف با داشتن مقدار درجه و یا دقیقه می توان فاصله از خط استوا و خط نصف النهار را بدست آورد.

۱-۱-٤ تبدیل دقیقه به درجه

برای تبدیل مثلا ۳۵۴۲.۹۶۷۰۲ درجه باید به صورت زیر عمل کنیم:

این عدد نشان دهنده ی ۳۵ درجه و ۴۲.۹۶۷۰۲ دقیقه می باشد .

که معادل درجه اش میشود : ۳۵+ ۶۰/(۴۲.۹۶۷۰۲) = درجه۳۵.۷۱۶۱۱۷

یعنی دقیقه را باید تقسیم به ۶۰ کرده و با درج جمع کنیم.

۱-۱-۵ مرکز مختصات

محل تلاقی مدار استوا و مدار نصف النهار (سمت گرینویچ) مبدا مختصات gps در طول و عرض جغرافیای می باشد در ارتفاع هم سطح دریا مبدا می باشد برای مثال در مورد عرض جغرافیایی: صفحه استوا را در نظر بگیرید، بالای این صفحه را (روی کره)با ۱۸۰ صفحه هم محور (محور گذرنده از مرکز زمین و عمود بر صفحه نصف النهار مبدا) و هم فاصله (زاویه) قطع دهید .(دقت ۱ درجه) . حال اگر یک نقطه روی صفحه ی ۲۵ ام باشد عرض جغرافیایی آن نقطه ۲۵ درجه می شود .برای بقیه نقاط هم به همین صورت می باشد.

۱-۱-٦ ویژگی های GPS canmore

دو تا کانال برای استفاده کردن دارد

۳۲ track erificiation channel

Tracking sensitivity-۱۶۲dBm(دقت گیرندگیه جی پی اس هست که هر چه منفی تر باشه بهتره)

به خاطر track verification ۳۲ و ۲ تا ورودی دریافت سریع ماهواره و زمان شروع را کاهش می دهد.حساسیت ورودی ۱۴۷dBm و حساسیت ردیابی ۱۶۳dbm که کارایی خوبی برای نویگیشن حتی در محیط نسبتا بسته ایجاد می کند.

سیستم های تقویت بر مبنای ماهواره مانند WAAS وEGNOs ساپرت می شوند که دقت را افزایش دهند.علاوه بر آن تابع SAGPS را هم ساپرت می کند .

اینترفیس سریال RS۲۳۲ بر روی یک اینترفیس اتصال دهنده قرار دارد.

ولتاژ تغذیه ی آن ۳.۳ تا ۶ ولت است.

کاربرها می توانند جملات NMEA یا کدهای باینری با اضافه کردن فلش مموری تغییر بدهند.

پروتکل ارتباطی که این جی پی اس استفاده می کند emea است

۱-۱-٦ پیام های NMEA

فرمت GGA

GGA-(GP,GL,GN)GNSS DOP and active satellite

اطلاعاتی که می دهد در ادامه توضیح می دهد:

عدد روبروی *Hdop* معادل دقت دستگاه شما در تعیین طول و عرض جغرافیایی است

فرمت پیامی که میدهد به صورت زیر می باشد:

$GPGGA,<۱>,<۲>,<۳>,<۴>,<۵>,<۶>,<۷>,<۸>,<۹>,M,<۱۰>,M,<۱۱>,<۱۲>,*<۱۳><CR><LF>

یک مثال از این فرمت

Example:
$GPGGA,104549.04,2447.2038,N,12100.4990,E,1,06,01.7,00078.8,M,0016.3,M,,*5C<CR><LF>

Field	Example	Description
1	104549.04	UTC time in hhmmss.ss format, 000000.00 ~ 235959.99
2	2447.2038	Latitude in ddmm.mmmm format Leading zeros transmitted
3	N	Latitude hemisphere indicator, 'N' = North, 'S' = South
4	12100.4990	Longitude in dddmm.mmmm format Leading zeros transmitted
5	E	Longitude hemisphere indicator, 'E' = East, 'W' = West
6	1	Position fix quality indicator 0: position fix unavailable 1: valid position fix, SPS mode 2: valid position fix, differential GPS mode
7	06	Number of satellites in use, 00 ~ 12
8	01.7	Horizontal dilution of precision, 00.0 ~ 99.9
9	00078.8	Antenna height above/below mean sea level, -9999.9 ~ 17999.9
10	0016.3	Geoidal height, -999.9 ~ 9999.9
11		Age of DGPS data since last valid RTCM transmission in xxx format (seconds) NULL when DGPS not used
12		Differential reference station ID, 0000 ~ 1023 NULL when DGPS not used
13	5C	Checksum

ابتدای پیام با دلار شروع می شه و با خواندن چند کاراکتر اول مشخص می شود که فرمت پیام فرستاده شده چه می باشد اگر GGA بود که فیلد دوم عرض جغرافیایی و فیلد چهارم طول جغرافیایی می باشد و این یه پیام به شکل string است که با دلار شروع می شه و با ستاره پایان پیام می باشد بعد از ستاره دو تا char آورده که checksum می باشد و به شکل هگزا دسیمال می باشد که xor کاراکترهای بین دلار و ستاره می باشد.

همان طور که مشاهده می شود در فیلد دوم عرض جغرافیایی به فرمت ddmm.mmmm می باشد که dd درجه و mm.mmmmدقیقه می باشد که از روش بالا می تونی کاملا به درجه یا کاملا به دقیقه یا فاصله تبدیل کرد.

http://www.csgnetwork.com/gpscoordconv.html

سایتیه که تبدیل رو انجام می ده به وسیله ی این سایت می توان بررسی کرد که آیا درست تبدیل را انجام دادید یا نه

فیلد ۳: شمالی یا جنوبی را مشخص می کند که در قسمت کار ما مهم نیست مطمئنا از اینجا تا جلو در حافظ شمال و جنوب جغرافیایی تغییر نمی کند.

فیلد ۴:

طول جغرافیایی است که به فرمت dddmm.mmmm می باشد مثل عرض جغرافیایی می توان آن را از روش بالا حساب کرد.

اطلاعاتی که در بقیه ی فرمت ها نیاز داریم نیز همین ها می باشد که در ادامه دیگر فرمت ها آورده می شود و طرز کار با هر کدام از آن ها نیز به صورت بالا می باشد.

بقیه ی فرمت ها نیز اطلاعات طول وعرض جغرافیایی را می دهند در زیر هر کدازم از فرمت ها با یک مثال آورده شده است:

فرمت GLL:

طول و عرض جغرافیای موقعیت فعلی و زمان و حالت را می دهد(بالایی نوشته بود موقعیت را می دهد فرقشو با این موقعیت نمی دونم! اساسا نباید فرق کنند جفتشون باید طول و عرض جغرافیایی مکان فعلی رو بدن تو بالایی ولی دقیقا ذکر نکرده بود اینجا کرده بود.)

Format:
$GPGLL,<1>,<2>,<3>,<4>,<5>,<6>,<7>*<8><CR><LF>

Example:
$GPGLL,2447.2073,N,12100.5022,E,104548.04,A,A*65<CR><LF>

Field	Example	Description
1	2447.2073	Latitude in ddmm.mmmm format Leading zeros transmitted
2	N	Latitude hemisphere indicator, 'N' = North, 'S' = South
3	12100.5022	Longitude in dddmm.mmmm format Leading zeros transmitted
4	E	Longitude hemisphere indicator, 'E' = East, 'W' = West
5	104548.04	UTC time in hhmmss.ss format, 000000.00 ~ 235959.99
6	A	Status, 'A' = valid position, 'V' = navigation receiver warning
7	A	Mode indicator 'N' = Data invalid 'A' = Autonomous 'D' = Differential 'E' = Estimated
8	65	Checksum

که همان طور که مشخص است فیلد ش طول جغرافیایی در فرمت ddmm.mmmm را می دهد و فیلد ۳ عرض جغرافیایی در فرمت dddmm.mmmm

فرمت GSA

این فرمت مثل اینکه جی پی اس را در حالت ریسیور قرار می دهد و ماهواره ها و مقادیر DOP برای راهبری استفاده می شود.

فرمت GSV نیز برای خوندن اطلاعات ماهواره استفاده می شه که ماهواره در چه زاویه ای قرار داردو...به درد ما نمی خورد!

Format:
$GPGSA,<1>,<2>,<3>,<3>,<3>,<3>,<3>,<3>,<3>,<3>,<3>,<3>,<3>,<3>,<4>,<5>,<6>*<7><CR><LF>

Example:
$GPGSA,A,3,26,21,,,09,17,,,,,,,10.8,02.1,10.6*07<CR><LF>

Field	Example	Description
1	A	Mode, 'M' = Manual, 'A' = Automatic
2	3	Fix type, 1 = not available, 2 = 2D fix, 3 = 3D fix
3	26,21,,,09,17,,,,,,	PRN number, 01 to 32, of satellite used in solution, up to 12 transmitted
4	10.8	Position dilution of precision, 00.0 to 99.9
5	02.1	Horizontal dilution of precision, 00.0 to 99.9
6	10.6	Vertical dilution of precision, 00.0 to 99.9
7	07	Checksum

به نظر کاملترین frame مروبط به پروتکل GPRMC است ولی به نظرم کوتاه ترین پروتکل انتخاب بشه بهتره برای ما چون فقط ما طول و عرض جغرافیاییشو می خوایم

$GPRMC,hhmmss.ss,A,llll.ll,a,yyyyy.yy,a,x.x,x.x,dd mmyy,x.x,a*hh

۱- زمان را به وقت گرینویچ با دقت ۱۰۰ ام ثانیه نشان می دهد.

۲-وضعیت دیتا را نشان می دهد(V= warning , A= Success)

۳-عرض جغرافیایی را نشان می دهد.

۴-شمال و جنوب را نشان می دهد (n,s)

۵- طول جغرافیایی را نشان می دهد.

۶- شرق یا غرب را نشان می دهد (w,s)

۷- سرعت روی زمین بر حسب گره ی دریایی را می دهد.

۸-زاویه ی مداری که اطلاعات از آن می آید را به ما می دهد.

۹-تاریخ را به میلادی به ما می دهد.

۱۰- تغییرات مغناطیسی زمین را بر حسب درجه نشان می دهد.

۱۱-شرق و غرب مغناطیسی را می دهد.

Format:
$GPRMC,<1>,<2>,<3>,<4>,<5>,<6>,<7>,<8>,<9>,<10>,<11>,<12>*<13><CR><LF>

Example:
$GPRMC,104549.04,A,2447.2038,N,12100.4990,E,016.0,221.0,250304,003.3,W,A*22<CR><LF>

Field	Example	Description
1	104549.04	UTC time in hhmmss.ss format, 000000.00 ~ 235959.99
2	A	Status, 'V' = navigation receiver warning, 'A' = valid position
3	2447.2038	Latitude in dddmm.mmmm format Leading zeros transmitted
4	N	Latitude hemisphere indicator, 'N' = North, 'S' = South
5	12100.4990	Longitude in dddmm.mmmm format Leading zeros transmitted
6	E	Longitude hemisphere indicator, 'E' = East, 'W' = West
7	016.0	Speed over ground, 000.0 ~ 999.9 knots
8	221.0	Course over ground, 000.0 ~ 359.9 degrees
9	250304	UTC date of position fix, ddmmyy format
10	003.3	Magnetic variation, 000.0 ~ 180.0 degrees
11	W	Magnetic variation direction, 'E' = East, 'W' = West
12	A	Mode indicator 'N' = Data invalid 'A' = Autonomous 'D' = Differential 'E' = Estimated
13	22	Checksum

بقیه ی فرمت ها به درد کار ما نمی خورد از آوردنش خود داری کردم!

۱-۱-۷ اتصال gps به لپ تاپ

پس از لحیم کاری پایه های جی پی اس به کابل سریال به لپ تاپ وصل کرده و بااستفاده از کد c# ارتباط با پورت سریال برقرار شده و با استفاده از کتابخانه های موجود در c# اطلاعات خوانده شده در یک فایل ذخیره شده که در ادامه اطلاعات آورده شده است

خواندن اطلاعات GPS با استفاده از نرم افزار های GPSSports_installer_pro و GPSviewer که اطلاعات خوانده شده به وسیله ی این نرم افزار ها در طی سه آزمایش حرکت از درب دانشکده تا در حافظ در کاغذ نوشته شده است.

۱-۱-۸ نمایش اطلاعات دریافت شده از gps در google map و بررسی مکان های بدست آمده

اطلاعات بدست آمده در ابتدا در یک فایل excel قرار داده شده و برای این که بتوان فرمت اطلاعات بدست آمده از gps به اطلاعات مورد استفاده در google maps تبدیل کرد فرمت dddmm.mmmm از جی پی اس بدست می آید که با تقسیم mm.mmmm به ۶۰ و اضافه کردن آن به ddd فرمت مورد استفاده در گوگل مپ بدست می آید با این کار در excel اطلاعات مورد نظر بدست آمد

اطلاعات حاصل به صورت lattitudeدر ستون اول longitude در ستون دوم و صفر در ستون سوم در excel به فرمت CSV ذخیره کرده و سپس با استفاده از فایل notpad آن را باز کرده مشاهده می شود اطلاعات حاصل به صورت ستونی از lat و lon و کاما در بین آن ها ذخیره شده است .حال باید این اطلاعات را به صورت کد html به google map بدهیم کد html زده شده :

<?"٨" encoding="UTF-١٠٠xml version="?>

<"٢.٢kml xmlns="http://earth.google.com/kml/>

<Document>

<name>Paths</name>

<description> Point at the height of the underlying terrain.</description>

<"Style id="yellowLineGreenPoly>

<LineStyle>

<ffff</color٠٠fvcolor>>

<</width٣width>>

<LineStyle/>

<PolyStyle>

<</color٠٠ff٠٠fvcolor>>

<PolyStyle/>

<Style/>

<Placemark>

<name>Example placemark</name>

<description>Point at the height of the underlying terrain.</description>

<styleUrl>#yellowLineGreenPoly</styleUrl>

<LineString>

<</extrude١٠extrude>>

<</tessellate١٠tessellate>>

<altitudeMode>clampToGround</altitudeMode>

<coordinates>

اطلاعاتی که از جی پی اس بدست آمده و در فایل CSV ذخیره شده در اینجا کپی می کنیم.

<coordinates/>

<LineString/>

<Placemark/>

<Document/>

<kml/>

فایل را به فرمت kml ذخیره کرده و سپس آن را با گوگل مپ باز می کنیم مشاهده می کنیم که مسیری که مختصات جغرافیایی آن را بدست آورده بودیم در نرم افزار به صورت یک خط زرد دیده می شود:

با مشاهده ی مسیرهای بدست آمده که فایل های آن در فایل ضمیمه ی ۱ قرار داده شده است مشاهده می شود که میزان خطای حاصل از gps زیاد می باشد و به تنهایی قادر به هدایت ربات نخواهد بود به همین دلیل از ترکیب اطلاعات آن با ادومتری سعی شده است محل دقیق چرخش بدست بیاید و با استفده از سنسور Xscense جهت حرکت ربات در هر حالت کنترل می شود که در ادامه با جزئیات آن شرح داده می شود.

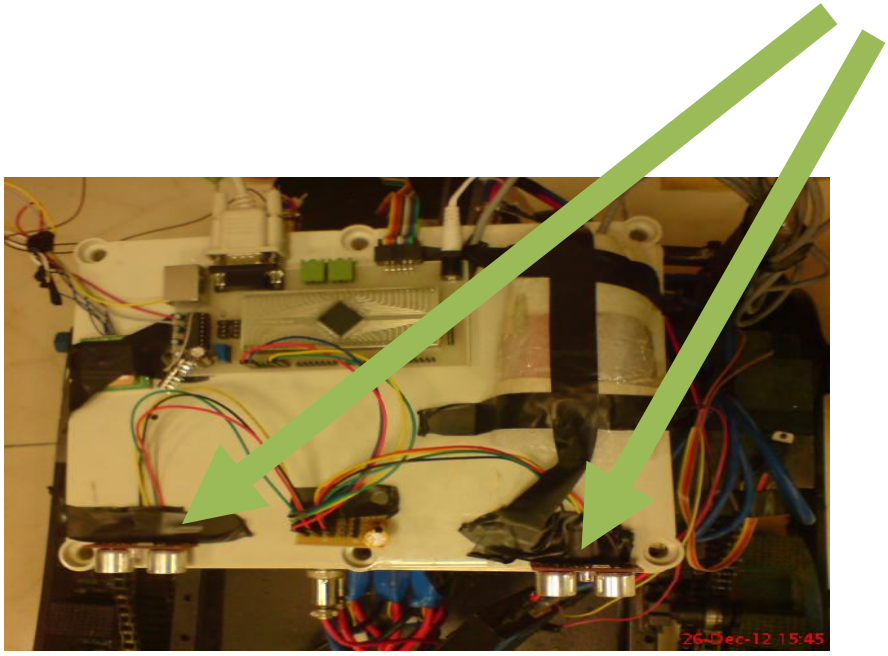
تست های گرفته شده از جی پی اس در فایل ضمیمه ی ۲ قرار داده شده است.

Ultrasonic ۲-۱

از این سنسور برای تشخیص موانع استفاده شده است و چون در این مسیر فرض بر این است که ربات با استفاده از سنسورهای دیگر در مسیر مناسب حرکت کرده پس موانع ایجاد شده فقط انسانی می باشد که در این صورت ربات می ایستد تا انسان از جلوی آن کنار برود برای این کار دو سنسور در جلوی ربات قرار داده شده است.

اطلاعات سنسور با استفاده از پورت سریال از طریق کد C# به صورت مجزا خوانده شده است.

اولتراسونیک ها یی که برای مشاهده مانع در جلوی ربات قرار داده شده اند



۱-۲-۱ راه اندازی ماژول srf۰۸

ولتاژ کاری این ماژول ۵+ ولت می باشد و در حالت کاری جریان مصرفی آن ۱۵ mAمی باشد.
و می تواند فاصله ربات را با اجسامی که در فاصله ۳ سانتی متر تا ۶ متری هستند را اندازه گیری کند .برای راه اندازی این فاصله سنج (SRF۰۸)از **پروتکل ارتباطی i۲c** استفاده می نماییم.
این ماژول دارای ۵ پایه (Power / SDA / SCL / Do Not Connect / GND) می باشد.
که مطابق شکل پایه Power به ۵+ و پایه ی GND را به زمین متصل می نماییم.
همانطور که از اسم پایه Do Not Connect مشخص می باشد این پایه را **نباید** به جایی متصل نماییم.
و در پایان پایه های SCL , SDA را توسط ۲ تا مقاومت ۱ kاهم pullup می کنیم.
دقت داشته باشید که پایه Power , GND را اشتباه متصل نکنید.

RSF۰۸ به صورت مجموعه ای از ۳۶ رجیستر نشان داده میشود.

Location	Read	Write
0	Software Revision	Command Register
1	Light Sensor	Max Gain Register (default 31)
2	1st Echo High Byte	Range Register (default 255)
3	1st Echo Low Byte	N/A
~~	~~~~~	~~~~~
34	17th Echo High Byte	N/A
35	17th Echo Low Byte	N/A

فقط در ۰،۱،۲ location میتونان نوشت.

۰Location رجیستر دستور(command register) است و برای شروع مرحله Ranging استفاده میشود. خواندن از ۰locaton ورژن نرم افزار SRF۰۸ را میدهد. به صورت پیش فرض Ranging بعد از ۶۵ms پایان میابد، ولی این میتواند تغییر کند با نوشتن در رجیستر ۲location. با انجام اینکار باید AnalogueGain در ۱ locationتغییر داده شود.

۱ Locationسنسور نور پردازنده است. این دیتا هرزمان که یک دستور Ranging تکمیل میشود update میشود. و زمانیکه دیتای Rang خوانده شود میتواند خوانده شود(؟and can be read when range data is read). دو مکان بعدی،۲و۳، ۱۶ بیت دیتای بدون علامت، نتیجه آخرین Ranging است–بایت پر ارزش ابتدا. معنی این مقدار به دستور استفاده شده وابسته است. میتواند فاصله به اینچ،یا سانتیمتر، یا زمان flight به uS باشد. مقدار صفر نشان دهنده ی آن است که هیچ شیئی یافت نشده است.

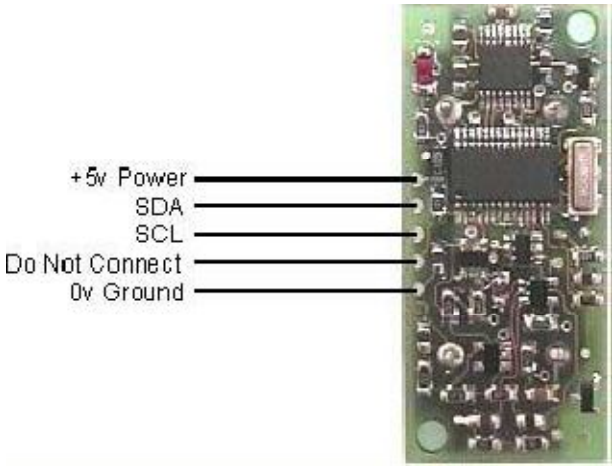
۱-۲-۲ دستور ها

سه دستور برای شروع تعیین مسافت (Ranging) موجود است، (۸۰ تا ۸۲)که برای برگرداندن پاسخ با فرمت اینچ، سانتیمتر، یا میکرو ثانیه استفاده میشود. چند دستور برای حالت Arfificial Neural Network(ANN) که ما نیازی به آن نداریم. و چند دستور دیگر برای تغییر آدرس I۲C.

Command		Action
Decimal	Hex	
80	0x50	Ranging Mode - Result in inches
81	0x51	Ranging Mode - Result in centimeters
82	0x52	Ranging Mode - Result in micro-seconds
83	0x53	ANN Mode - Result in inches
84	0x54	ANN Mode - Result in centimeters
85	0x55	ANN Mode - Result in micro-seconds
160	0xA0	1st in sequence to change I2C address
165	0xA5	3rd in sequence to change I2C address
170	0xAA	2nd in sequence to change I2C address

۱-۲-۳ مود فاصله سنجی(Ranging Mode)

برای شروع(initiate) مود فاصله سنجی، یکی از دستورات بالا را در رجیستر دستور مینویسیم. و به میزان مورد نیاز صبر میکنیم تا عملیات تکمیل شود، سپس هر مقدار پاسخ و ریزالتی که میخواهیم میخوانیم. بافر echo در ابتدای هر ranging پاک میشود. اولین echo rang در location۲,۳ قرار داده میشود. ثانیه در ۴و۵ و ... . اگر Location، (highe and low bytes) صفر بود، دیگر خواندن بقیق رجیستر ها لازم نیست. زمان پیشنهادی و پیش فرض برای اتمام ranging۶۵ ms میباشد. با اینحال میتوانید آن را کمتر کنید با نوشتن در range register قبل از اجرای Ranging command. دیتای سنسور نور در location۱ بعد از دستور ranging به روز رسانی میشود.



IMU ۳-۱

یک imu با کشف سرعت فعلی شتاب با به کار بردن یک یا بیشتر از شتاب سنج و تشخیص تغییر در رفتار چرخشی مانند pitch and roll and yaw با استفاده از یک یا تعداد بیشتری gyroscope

Imu یک واحد مجرد در ماژول الکترونیک است که سرعت زاویه ای و شتاب خطی را جمع می کند که به main priccessor فرستاده شده است.

محفظه ی imu از دو سنسور مجزا تشکیل شده است سنسور اول یک مجموعه ی سه تایی از شتاب سنج هاست که سه سیگنال آنالوگ ایجادمی کند که شتاب را در طول سه محور که باعث حرکت وسیله شده است را مشخص می کند.

به خاطر فشار سیستم و محدودیت های فیزیکی قسمت مهمی ازاین شتاب دریافتی ناشی از شتاب زمین می باشد.

سنسور دوم سنسور چرخش زاویه ای که متشکل از سه سنسور می باشد،می باشد.و همچنین سه سیگنال آنالوگ به خروجی می دهد.این سیگنال ها سرعت زاویه ای وسیله ی نقلیه را در راستای سه محور می دهد.با وجود اینکه سنسور imuد رمرکز جرم وسیله قرار ندارد اندازه ی سرعت زاویه ای تحت تاثیر شتاب زاویه ای یا خطی قرار نخواهد گرفت.

داده ها ی گرفته شده از این سنسورها به وسیله ی IMU ۶۸۱۱ microprocessor جمع می شود از بین یک برد۱۲ بیتیه ADC.اطلاعات سنسور سپس به پروسسور اصلی به وسیله ی .serial communications interface at a rate of about ۲۰۰ Hz RS۴۲۲باز گردانده می شود.

دستگاه مختصاتی که بر روی آنها سنسور های شتاب سنج و سنسور های سرعت زاویه ای طوری قرار میگیرند که با محورهای مختصات خود وسیله هم راستا قرار نگیرد.این به این خاطر است که دو سنسور در imu در دو جهت متفاوت در محفظه سوار شده اند که با دستگاه مختصات خود وسیله هم راستا نیستند.

Compass zcc۲۱۰ ٤-۱

سنسور استفاده شده در این بخش با نام zcc۲۱۰ n است که به دو روش می توان این سنسور را مورد استفاده قرار داد :

روش اول استفاده از ارتباط i۲c

روش دوم استفاده از ارتباط rs۲۳۲ یا همان سریال

که در این دو روش تنها تفاوتی که وجود دارد نحوه اتصال این سنسور به میکرو است .

در این بخش به راه اندازی این سنسور با ارتباط i۲c می پردازیم:

نحوه اتصال این سنسور به میکرو:

پایه هایی که در این ارتباط استفاده می شودgnd, vcc , scl , sdaاست . ولتاژ تغذیه این سنسور ۵ ولت است. اما در صورتی که بخواهید این سنسور را با ولتاژ های ۶ تا ۹ ولت تغذیه کنید بجای استفاده از پایهvccباید ازvddاستفاده کنید. و پایهvccرا آزاد قرار دهید.

اطلاعات رجیستر ها :

با توجه به دیتا شیتی که این سنسور دارد آدرس رجیستری که این سنسور در هنگام نوشتن دارد برابر با عدد ۴۲ در مبنای هگز و هنگام خواندن برابر ۴۳ در مبنای هگز است .

این سنسور دارای ۲ مد کاری است که مد اول Single Sampling نام دارد و مد دوم با نام Consecutive collection شنا خته می شود .

در مد اول سنسور یک بار از زاویه نمونه برداری کره و آن را ذخیره می کند. و در مد دوم سنسور هر یک ثانیه یک بار از زاویه نمونه برداری کرده و آن را ذخیره می کند.

برای ورد به مد اول کافی است که عدد ۷۴ و برای ورود به مد دوم کافیست که عدد ۷۶ در مبنای هگز زا ارسال کرد.

خواندن اطلاعات در هنگام استفاده از این دو مود با ارسال عدد ۷۷ در مبنای هگز برای ماژول انجام می شود. مازول با دریافت این کد اطلاعات پردازش شده را در قالب دو بیت برای میکرو ارسال می کند.

برنامه:

برنامه نوشته شده برای میکرو کنترلر atmega۱۶ و کامپایلر cod vision است.

```
i۲c_start();
i۲c_write(۰X۴۲);
i۲c_write(۰X۷۴);
i۲c_stop();
while (۱)
{


delay_ms(۵۰);

i۲c_start();
i۲c_write(۰X۴۲);
i۲c_write(۰X۷۷);
i۲c_stop();
delay_ms(۲۰);

i۲c_start();
i۲c_write(۰X۴۳);
```

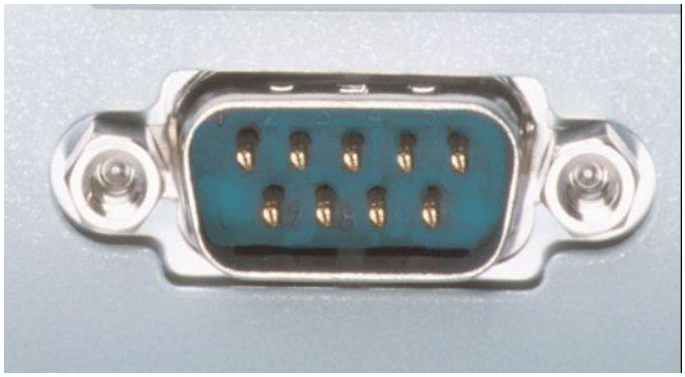
```
y=i۲c_read(۱);
w=i۲c_read(۰);
i۲c_stop();

lcd_clear();
itoa(w,x);
lcd_putsf("x");
lcd_puts(x);

}
```

اگر در نوشتن این برنامه یا راه اندازی سنسور به مشکلی برخوردید حتما مشکلات رو با ما در میان بزارید.

۲ کار با نرم افزار ها

۲-۱ طریقه ی ارتباط با پورت سریال

سریال پورت استاندارد ۹ پین دارد که اطلاعات را بیت به بیت ارسال و دریافت میکند .هر یک از پین های آن وظیفه ای دارند:



شماره پین	کاربرد
۱	علامت تشخیص تماس
۲	دریافت اطلاعات
۳	ارسال اطلاعات
۴	آمادگي ترمينال
۵	اتصال زمین
۶	آماده باش
۷	آماده ارسال
۸	آماده دریافت
۹	نشانگر زنگ تلفن(کاربرد در مودم)

۲-۲ کار با سریال پورت به وسیله ی #c:

۱- کتابخانه ی using System.IO.Ports; را فراخوانی کرده

۲-کنترل Serial Port ازTool box به فرم برنامه اضافه می کنیم

۳-یک List box بنام lstPort را از Toolbox به فرم برنامه اضافه می کنیم

۴-در رویداد لود فرم کد زیر را برای بدست آوردن لیست پورتهای کام سیستم وارد کردیم

```
string[] ports = SerialPort.GetPortNames();
lstPort.Items.AddRange(ports);
```

برای مشاهده پورت های کام سیستم به مسیر Control Panel --> Device Manager ---> LPT&COM Port می رویم.
۵-یک Button بنام btnOpen از Toolbox به فرم برنامه اضافه کرده و در رویداد کلیک آن کد زیر را وارد کردیم:

```
serialPort۱.PortName = lstPort.SelectedItem.ToString();
serialPort۱.Open();
```

با کد بالا پورت انتخاب شده از لیست باکس باز می شود .
۶-یک Textbox و یک Button بنام های txtMSG و btnSend به فرم اضافه کرده .
۷-در رویداد کلیک btnSend کد زیر وارد شده است:

```
serialPort۱.WriteLine(txtMSG.Text);
```

کد بالا متن وارد شده در txtMSG را به پورت می فرستد . حالا برای خواندن از پورت در ادامه کد بالا کد زیر اضافه شد است:

```
MessageBox.Show(serialPort۱.ReadLine());
```

برای بستن پورت هم از کد زیر استفاده شده است:

```
serialPort۱.Close();
```

البته چون در لپ تاپ ها پورت کام وجود ندارد از یک نرم افزار پورت مجازی ساز مثل Virtual Serial Port Kit استفاده شده است که در ضمیمه ی ۴ قرار داده شده است.

۲-۳ طریقه ی ارتباط با میکرو

با استفاده از نرم افزار codevision که در ضمیمه ی ۳ قرار دارد کد لازم برای ارتباط با میکرو زده شده است که تمام اطلاعات با استفاده از پورت سریال از میکرو خوانده می شود و به صورت یک رشته ای از اعداد به ترتیب longitute،latitude، فاصله ی مانع از سنسور سمت راست،فاصله ی مانع از سنسور سمت چپ،درجه ی ربات به برنامه داده می شود و بر اساس این اطلاعات دریافت شده برای حرکت ربات تصمیم گیری می شود و سپس دستوری که لازم است به ربات داده شود به صورت یک رشته با استفاده از دستور serialPort.write به پورت سریال فرستاده می شود.

۲-٤ استفاده از C# متصل به میکرو

۱-ابتدا برنامه TCP-com را باز می کنیم در ضمیمه ی ۵ قرار داده شده است. sourcefile:magicconverter

۲-بعد TCP-IP را نصب می کنیم.

۳–از قسمت connectorاسم پورتی که در لپ تاپمون قبلا نداشتیم و انتخاب می کنیم با این کار یک virtual com ایجاد می شود

۴–قسمت create virtual را تیک می زنیم.

۵–درقسمت آی پی کام آی پی روتر را می دهیم

۶–remot port می شه شماره ی پورتش اینجا ۴۰۰۱

۷–برنامه #c باز شود در information->autumn->c#

با دستور e.hashcode می تونی کاراکترهایی که در کیبورد وارد می شه را خواند.

۳ آزمایشات عملی

۳-۱ محاسبه ی میزان انحراف ربات در حرکت به سمت جلو

شرایط آزمایش:کف آزمایشگاه ،سرعت ۶۳

ربات در مسیر مستقیم انحراف به چپ دارد برای پیدا کردن میزان انحراف ربات ،ربات را از نقطه ی صفر با زاویه ی صفر حرکت داده در فاصله ی ۳.۶ متر ۱۳ درجه انحراف بداشه است یعنی ۳.۶ درجه بر متر انحراف دارد در هنگام بازگشت از همان مسیر دو درجه انحراف داشته است که انی نشان می دهد خطای ربات ثابت می باشد

نتیجه :نمی توان به حرکت در مسیر مستقیم ربات اعتماد کرد و حتما باید کنترلی برای حرکت ربات قرار داد که در ادامه می بینیم با استفاده از دقت IMU استفاده شده توانستیم مشکل حرکت ربات در مسیر مستقیم را حل کنیم.

۳-۲ پیدا کردن $\Delta\theta$ مناسب

$\Delta\theta$ میزان انحرافی است که به ربات اجازه می دهیم از زاویه ی اصلی منحرف شود

$\Delta\theta$ را در ابتدا عدد ۶ و θ را صفر قرار دادیم بعد از ۴ متر حرکت خطای حرکت ربات صفر شد

ربات در این حالت بازی زیادی از خود نشان نداد و به راحتی به زاویه مورد نظر رسید

هنگامی که زاویه را ۷۹ قرار دادیم سرعت حرکت ربات بالا بود در نتیجه زاویه را ندید و از آن عبور کرد در نتیجه تصمیم گرفتیم سرعت را تابعی از $\Delta\theta$ قرار دهیم تا با افزایش آن سرعت ربات نیز افزایش پیدا کند و با کاهش آن سرعت ربات کاهش پیدا کند

تابعی به نام processrotatespeed نوشته شد که در آن سرعت چرخ محاسبه می شود

$\Delta\theta$ را در اعداد ۳۰،۲۵،۱۵،۶ امتحان کردیم و در $\Delta\theta$ =۶ حرکت ربات خوب بود و هنگامی که اختلاف زاویه ربات و زاویه ی مورد نظر بزرگ تر ۵۰ بود سرعت چخش را ۳۵ و اگر بزرگ تر از ۲۵ بود سرعت را ۳۰ در غیر این صورت سرعت چرخش ر ا ۲۰ قرار دادیم .

نتیجه:اگر زاویه ی انحراف را زیاد قرار دهیم حرکت مستقیم ربات تحت تاثیر آن قرار گرفته و پس از طی مسیر زیاد ممکن است کج شود و اگر میزان انحراف را کم قرار دهیم ربات برای قرار گرفتن در زاویه ی درست خیلی تکان خورده و چون حرکت ربات به صورت استپ استپ می باشد ممکن است مدت زمان زیاد تری برای قرار گرفتن در بازه ی مورد نظر صرف می کند.

۳-۳ محاسبه ی طول و عرض جغرافیایی مناسب برای قرار گیری ربات در state های مختلف

برای حرکت ربات در state های مختلف (که در ادامه راجع به آن ها صحبت می شود)نیاز به دانستن مقدار طول و عرض جغرافیا ی موقعیت های مختلف می باشد از آن جایی که خطای جی پس اس حدود ۲.۵ متر بوده است سعی شد در هر موقعیت با استفاده از اطلاعاتی که از جی پی اس پس از تست های متعدد بدست آمد حدود حر کت ربات را بدست آورده که اگر ربات از این حدود تجاوز کرد یعنی حرکت آن اشتباه می باشد و باید خود را اصلاح کند.

مقادیری که برای طول وعرض جغرافیایی در نظر گرفته شده است در زیر آورده شده است:

مرحله ی اول:

در ابتدای نوشتن برنامه به دلیل بودن چاله در میان راه قرار بر نوشتن ۵ حالت برای حرکت ربات بود(البته در ادامه خواهیم دید که پس از آزمایش های انجام شده مشاهده شد ربات به آسانی می توانست از چاله بگذرد پس نیاز به حرکت L مانند ربات برای جلوگیری از حرکت ربات از روی چاله نیست) به این صورت که ربات در حالت اول latو lon جلوی درب دانشکده را به صورت محدوده داشته باشد و چون در ابتدا خودمان ربات را در آن جا قرار می دهیم نیاز به تست برای درست یا غلط بودن نیست ولی چون در کد ابتدایی می خواستیم تغییرات طول و عرض جغرافیایی را در جهت مناسب حرکت حساب کنیم محاسبه ی latو lon نیاز بود (البته در ورژن های بعدی فقط محدوده ی حرکت ربات را در latو lon مناسب حساب کردیم) در قسمت زیر latو lon نواحی مورد نیاز نوشته شده است:

درب دانشکده:

Lat=۳۵/۷۰۴۱۰۰ Lon=۵۱/۴۰۸۵۸۱۶۷

چرخش اول:

Lat=۳۵/۷۰۴۲۸۳ Lon=۵۱/۴۰۸۶۶۳

Lat=۳۵/۷۰۴۲۹۸ Lon=۵۱/۴۰۸۶۳۷

چرخش اصلی:

Lat=۳۵/۷۰۴۳۶۸ Lon=۵۱/۴۰۸۷۰۸

Lat=۳۵/۷۰۴۳۸۹ Lon=۵۱/۴۰۸۶۴۹

درب حافظ:

Lat=۳۵/۷۰۴۶۸۱ Lon=۵۱/۴۱۲۱۵۱

در ابتدا به علت تغییرات زیاد lat در مسیر حرکت از چرخش اصلی به سمت در حافظ قرار شد که این مسیر را ۴ حالت در نظر بگیریم و در هر حالت lat آن را به عنوان محدوده ای که ربات در آن حرکت کند قرار دهیم اما در ادامه به علت دقت بالای imu مشاهده شد که لازم به انجام چنین کاری نیست چون مسیر مستقیم می باشد و کافیست ربات از جایی که چرخیده است تا انتها را مستقیم حرکت کندکه با سنسور imu قوی که در اختیار داشتیم انجام چنین کاری عملی بود در نتیجه برای حالت حرکت ربات پس از چرخش به سمت درب حافظ فقط یک محدوده به جای ۴ محدوده در نظر گرفته شد.

چهار حالتی که در ابتدا برای حرکت مستقیم ربات به سمت در حافظ در نظر گرفته شده بود:

حالت ۱:

Lat=۳۵/۷۰۴۳۰۴ Lat=۳۵/۷۰۴۵۰۷

حالت ۲:

Lat=۳۵/۷۰۴۵۶۶ Lat=۳۵/۷۰۴۴۵۱

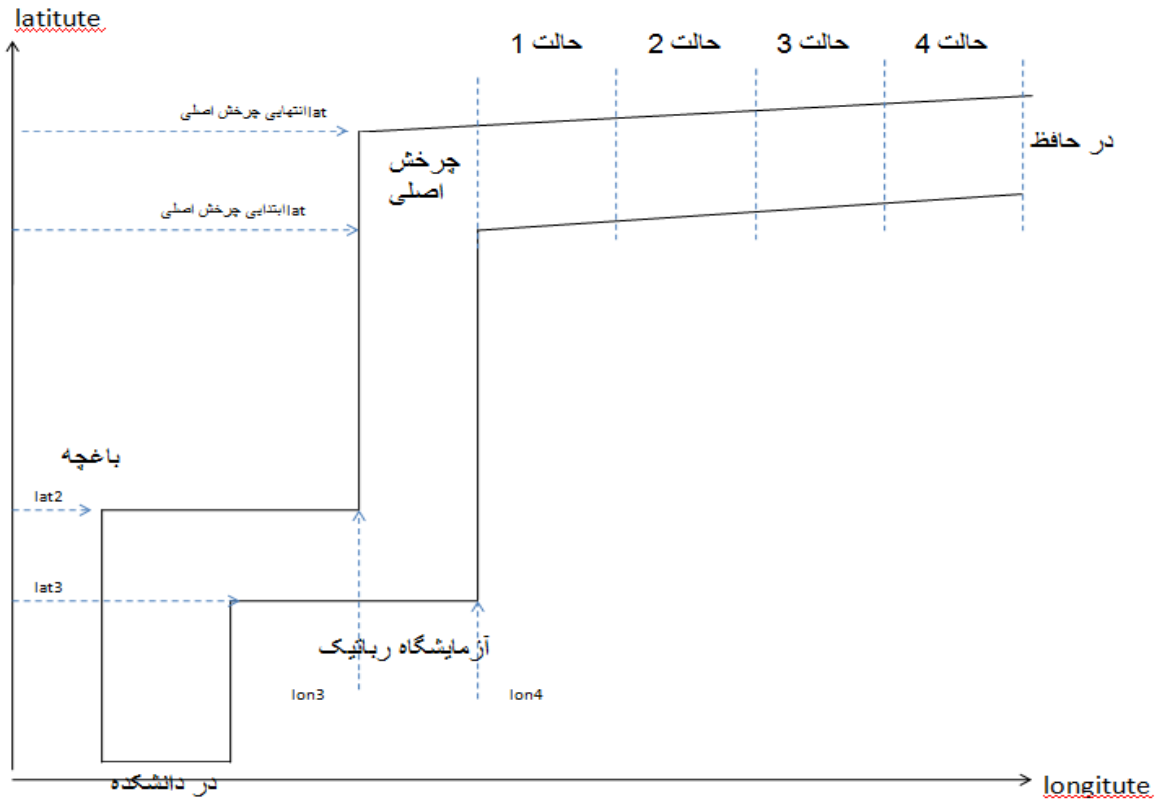
حالت۳:

Lat=۳۵/۷۰۴۶۳۸ Lat=۳۵/۷۰۴۴۸۸

حالت۴:

Lat=۳۵/۷۰۴۶۸۷ Lat=۳۵/۷۰۴۵۹۱

در این مسیر قرار بود به ربات گفته شود که به سمت lon افزایشی و در محدوده ی lat های مشخص شده ی بالا حرکت کند که البته همان طور که در بالا گفته شد در آخر برای آن مسیر یک بازه برای طول جغرافیایی در نظر گرفته شد و بقیه ی کار را به IMU واگذار کردیم.ضمن اینکه این ایده پیچدگی نوشتن کد و اجرا داشت پس باید سعی بر کم شدن حالت ها می کردیم



شکل۲:مسیری که در ابتدا برای حرکت ربات در نظر گرفته بودیم

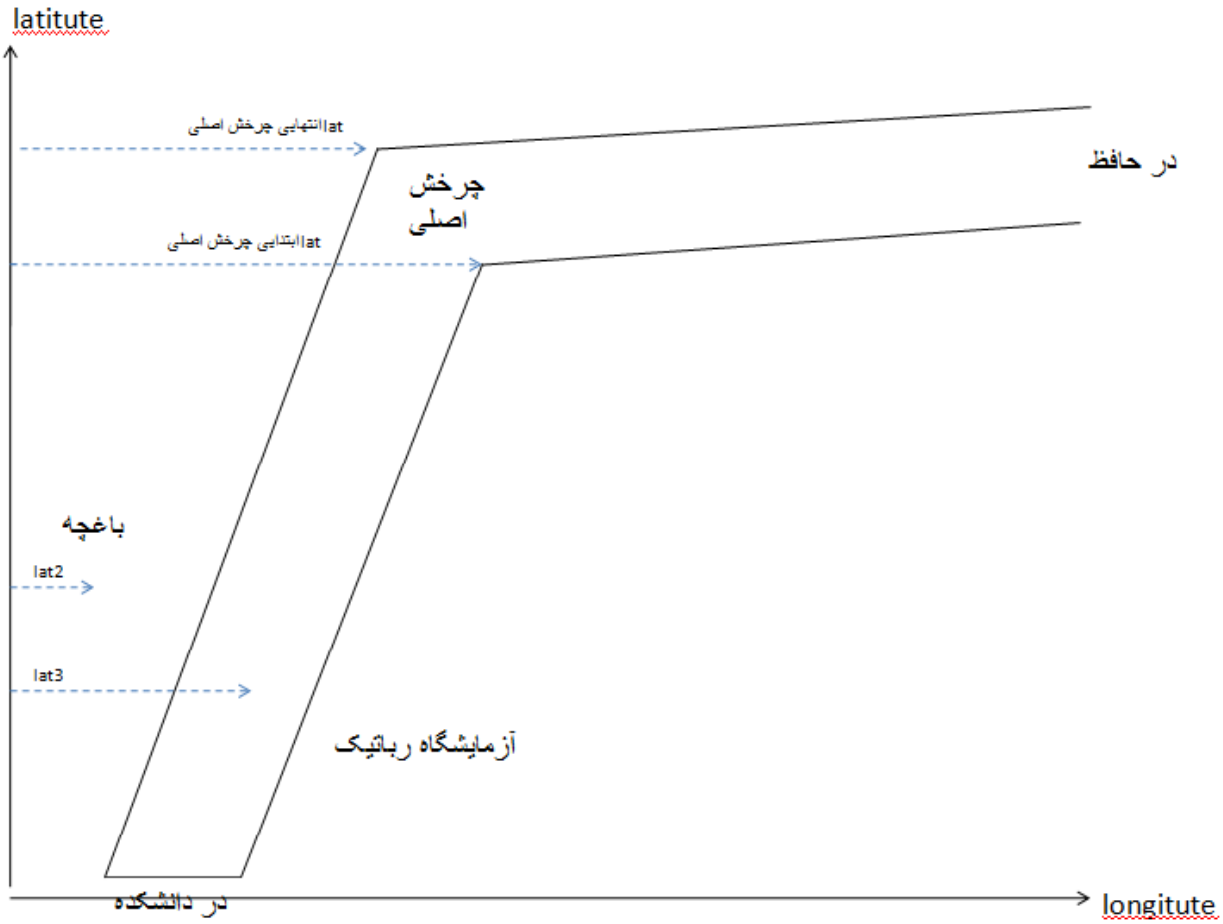
مرحله ی دوم:

در مرحله ی قبل مشاهده کردیم که چون مسیر حرکت به سمت در حافظ کاملا مستقیم نبود یعنی lat ثابتی نداشت تصمیم گرفته بودیم که مسیر حرکت را به چهار حالت تقسیم بندی کنیم که ربات به باغچه نرود ولی با بررسی های انجام شده تصمیم گرفتیم که این حالت ها را حذف کنیم و یک حالت به جای همه ی آن ها در نظر بگیریم در عوض با زاویه ای کمتر از ۹۰ درجه (۶۵)حرکت کنیم به علت دقت سنسور قطب نمای به کار گرفته شده مشاهده شد ربات تا حد خیلی خوبی در مسیر دلخواه حرکت می کند با این حساب فقط یک محدوده ی lat برای ربات در این حالت در نظر گرفته شد.

مرحله ی سوم:

چرخش اول و دوم حذف شد و تعداد حالات در کل به چهار کاهش پیدا کرد و علاوه بر جی پی اس از اطلاعات ادومتری برای پیدا کردن محل دقیق چرخش استفاده شد که با تست های

ایرادات روش:به علت خطای زیاد جی پی اس چرخش به موقع و درست ربات در چرخش ها کاری دشوار بود ضمن اینکه با تست های انجام شده مشخص شد ربات قادر به عبور از روی چاله می باشد و اگر ربات در زاویه مشخصی از درب دانشکده به سمت چرخش اصلی حرکت کند بدون نیاز به چرخش به چرخش اصلی می رسد که فقط لازم بود زاویه حرکت ربات به درستی کنترل شود که این کار نیز با imu امکان پذیر بود در نتیجه این طرح رد شد.



۳-٤ تعیین زاویه ی مورد نیاز برای حرکت ربات در جلوی درب دانشکده

۱-در ابتدا با زاویه -۱۴ حرکت کردیم که به جدول روبروی دانشکده برخورد کرد

۲-زاویه ی حرکت -۲۲ به جدول برخورد نکرد اما اگر با زاویه ی بیشتری حرکت می کردیم ضریب اطمینان حرکت ما بیشتر می شد

۳-زاویه ی شروع -۲۴ قرار داده شد که پس از آزمایش به نتیجه ی خوبی رسیدیم و البته در این حالت ربات از روی چاله رد می شد

۳-۵ محاسبه ی زمان برای استفاده از ادومتری

ربات را با زاویه -۲۴ در جلوی در دانشکده حرکت دادیم و زمان رسیدن ربات به محل مناسب برای چرخیدن ربات را اندازه گرفتیم این کار را ۲مرتبه انجام دادیم سپس با گرفتن زمان سیستم و محاسبه ی اختلاف آن با زمان شروع زمان مناسب برای چرخش ربات را حساب کردیم که البته در سه آزمایش به نتایج خیلی خوبی نرسیدیم یک بار نرسیده به محل مناسب چرخید و به جدول برخورد کرد دفعه بعدی نیز در جای مناسبی نچرخید و از جدول بالارفت در مرتبه ای دیگر اندکی دیر چرخید و زا میله های ریخته شده بر روی زمین بالا رفت .

نتیجه:با ادومتری به تنهایی نمی توانستیم به محل دقیقی برای چرخش دست پیدا کنیم پس نیاز به ترکیب اطلاعات جی پی اس و ادومتری داشتیم .

۳-٦ محاسبه ی زمان ایستادن مانع برای اصلاح کردن اطلاعات ادومتری

به علت اینکه محاسبه ی ادمتری ما بر اساس زمان حرکت ربات بود با ایستادن مانع در جلوی آن این زمان افزایش پیدا می کرد و لی در حقیقت ربات حرکت نمی کرد در نتیجه ربات زود تر از زمان مورد نظر به حالت چرخش می رسید برای حل این مشکل زمان ایستادن مانع در جلوی ربات را به زمان نهایی ربات برای چرخش اضافه کردیم.

۳-۷ شرط رسیدن ربات به در حافظ و ایستادن آن

به علت قرار گرفتن در حافظ در زیر سقف اطلاعات جی پی اس خطای زیادی پیدا می کرد پس باید علاوه بر آن راه دیگری برای فهمیدن اینکه ربات به در حافظ رسیده است و باید بایستد پیدا می کردیم برای این کاردر کد قسمتی را قرار دادیم که اگر مانع بیش از ۲۰ ثانیه ایستاد و حرکت نکرد پس مانع ثابت می باشد پس در حافظ می باشد که البته به علت محدودیت منابع (باتری) قادر به تست آن نبودیم اما در شرایط آزمایشگاهی درست اجرا می شد و فرمان ایستادن اجرا می شد.

٤ کالمن فیلتر

در این قسمت به علت خطای جی پی اس به طوری که در بعضی مواقع داده هایی پرت گرفته می شد از فیلتر استفاده شده است برای این کار از فرمول های داده شده برای فیلتر کالمن استفاده شد که در زیر آورده شده است :

$$\hat{x}_{k+1} = \hat{x}_k + K_{k+1}(z_{k+1} - \hat{x}_k)$$

$$K_{k+1} = \frac{\sigma_k^2}{\sigma_k^2 + \sigma_z^2}; \qquad \sigma_k^2 = \sigma_1^2 \quad ; \quad \sigma_z^2 = \sigma_2^2$$

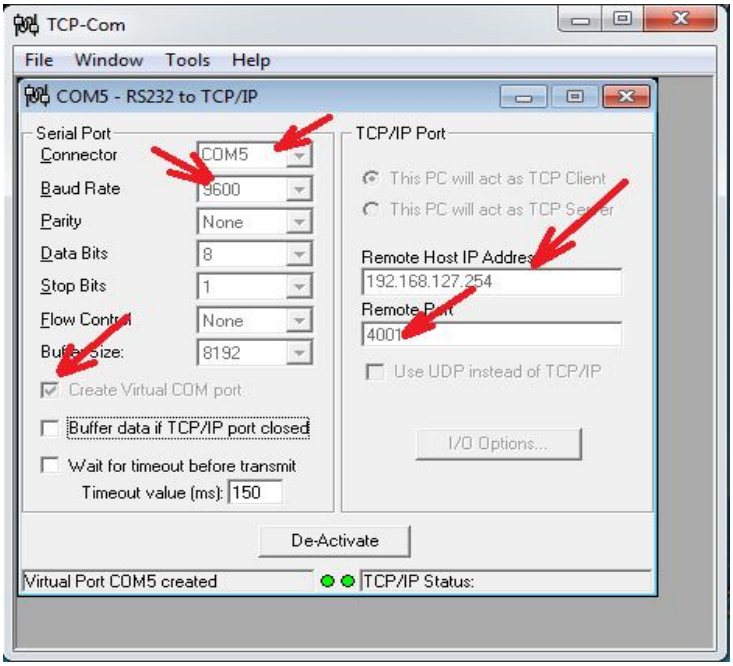
$$\sigma_{k+1}^2 = \sigma_k^2 - K_{K+1}\sigma_k^2$$

که در آن $\hat{x}_{k+1}$ مقدار به روز شده lat خوانده شده توسط GPS است. و z_{k+1} مقدار خوانده شده ی GPS و $\hat{x}_k$ مقدار آپدیت شده ی قبلی lat است. همچنین K_{k+1} ضریب محاسبه شده ی کالمن می باشد که به جای σ_z^2 مقداری ثابت قرار دادیم که با افزایش آن تاثیر داده ی خوانده شده ی فعلی افزایش پیدا می کند .

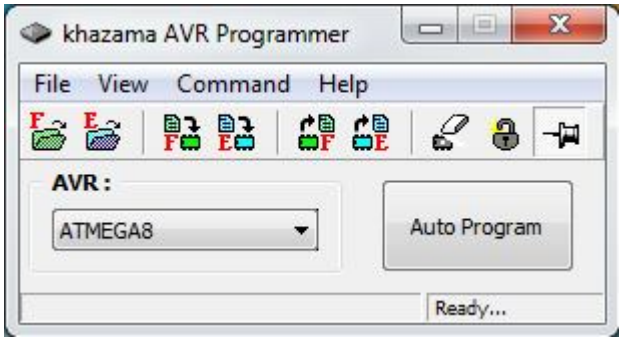
حال اگر مقدار خوانده شده ی lat توسط جی پی اس از lat مورد نظر ما برای چرخش بیشتر بود در آن صورت اطلاعات ادومتری ما نیز چک می شود که اگر به حد مورد نظر رسیده است موقع چرخش ربات می باشد از آن جایی که خروجی ادومتری در اکثر مواقع به علت قرار گرفتن مانع در جلوی آن یا قرار گرفتن در چاله یا تغییر زاویه بیشتر از حد مورد نظر داده می شد تقریبا در اکثر موارد هنگامی که جی پی اس موقعیت چرخش را مشخص می کرد ربات می چرخید.

۵ نرم افزارها

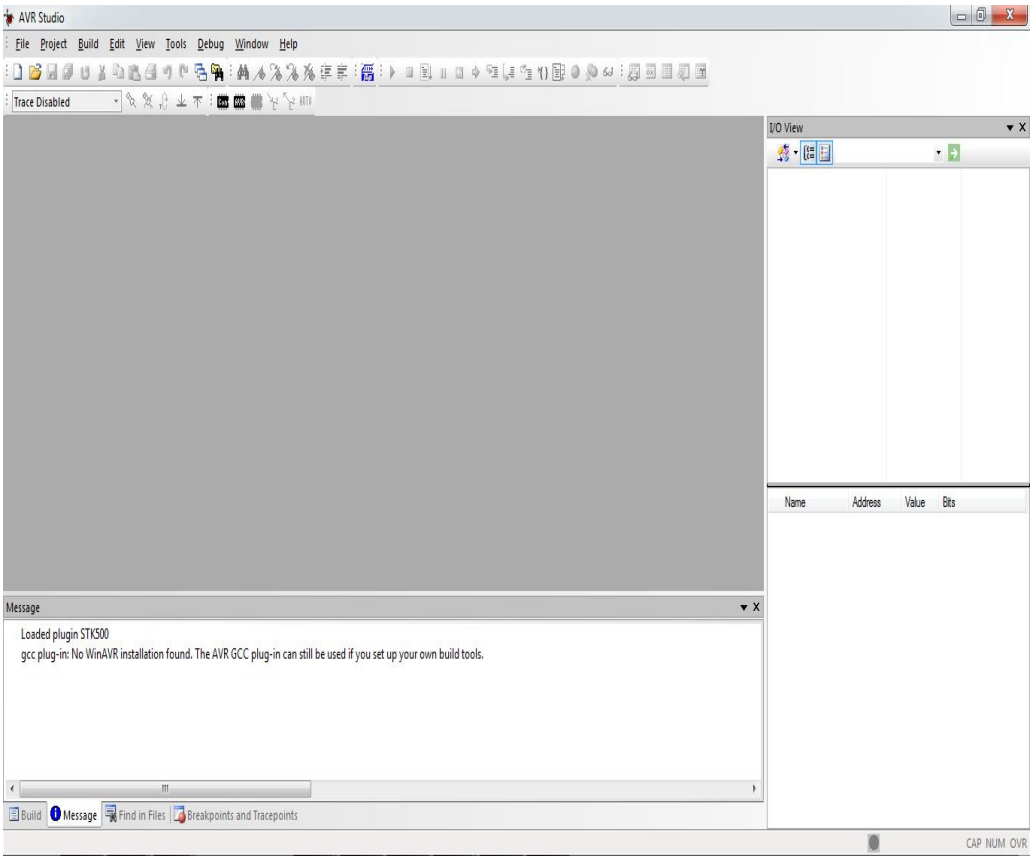
TCP-COM:



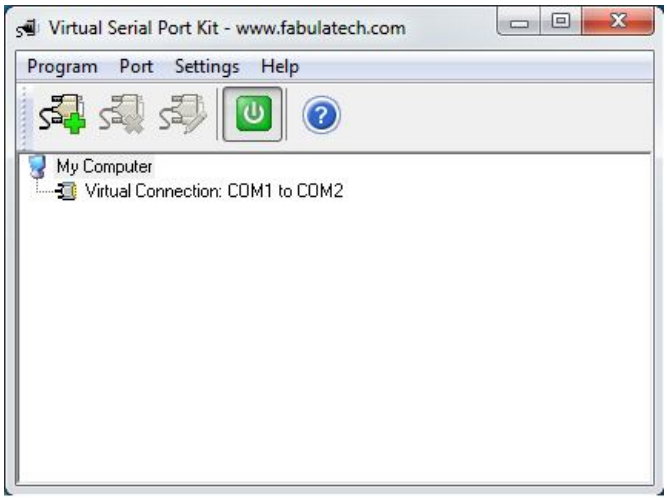
Khazarm AVR:



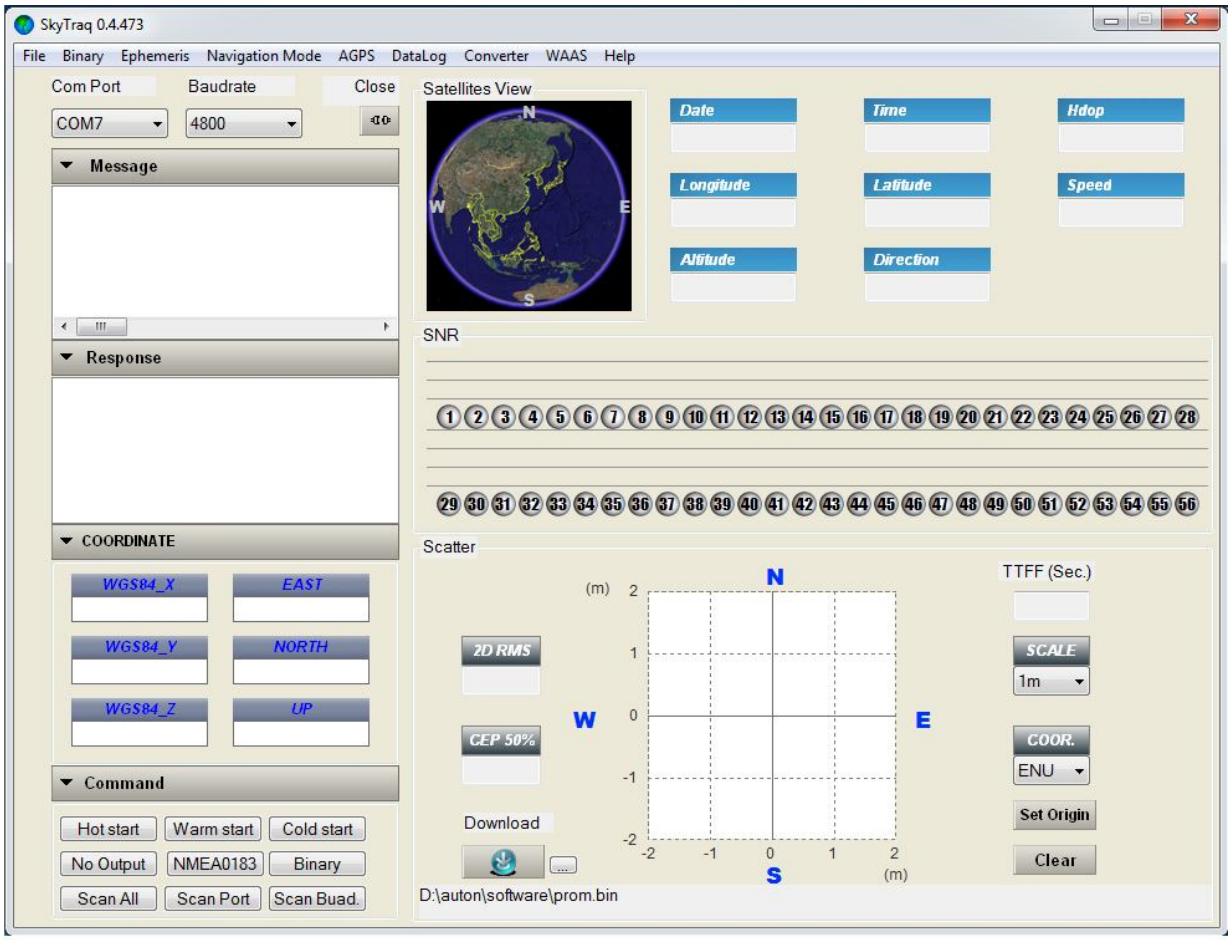
:AVRStudio



:Virtual Serial Port Kit



:GPS viewver


SkyTraq 0.4.473
File Binary Ephemeris Navigation Mode AGPS DataLog Converter WAAS Help
Com Port
Baudrate
Close
COM7
4800
Message
Response
COORDINATE
WGS84_X
EAST
WGS84_Y
NORTH
WGS84_Z
UP
Command
Hot start
Warm start
Cold start
No Output
NMEA0183
Binary
Scan All
Scan Port
Scan Buad.
Satellites View
N
W
E
S
Date
Time
Hdop
Longitude
Latitude
Speed
Altitude
Direction
SNR
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28
29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56
Scatter
(m)
2
1
0
-1
-2
-2 -1 0 1 2
(m)
N
W
E
S
2D RMS
CEP 50%
Download
TTFF (Sec.)
SCALE
1m
COOR.
ENU
Set Origin
Clear
D:\auton\software\prom.bin