آموزش طراحی و ساخت دست رباتیک آسمان

نویسنده: سلمان کریمی نسب

اردیبهشت ۹۴

فهرست

مقدمه

دست رباتیک "آسمان" به همت چند تن از دانشجویان مهندسی مکانیک عضو انجمن رباتیک دانشگاه شهید باهنر کرمان، طراحی و ساخته شده است. **پرهام اخلاص پور**، **علی فتحی زاده**، **مهراد ژند** و **رضا رواقی پاریزی** تحت سرپرستی **سلمان کریمی نسب** کارهایی نظیر طراحی و ساخت قسمت های مکانیکی، طراحی مدار و برنامه نویسی این ربات را بر عهده داشتند.

ربات مورد بحث در واقع مشابه دست راست انسان می باشد. این دست توانایی انجام بیشتر حرکات دست راست را دارد، این حرکات شامل: جمع و باز شدن دست ازمحل کتف، چرخش از محل کتف،جمع و باز شدن از محل آرنج ، پیچش مچ و حرکت مستقل ۵ انگشت دست می باشد. برای کنترل این ربات لباسی مجهز به سنسور های خمشی طراحی و ساخته شد. این لباس حرکات یک دست واقعی را ثبت میکند و پس از پردازش داده های ارسالی از سنسور های موجود در این لباس بازو حرکاتی مشابه با دست فردی که لباس را بر تن دارد انجام می دهد.

پروژه دست آسمان یک پروژه کاملا اپن سورس بوده و تمامی منابع آن شامل: طراحی سه بعدی، نقشه مدار های ربات، برنامه میکروکنترلر و آموزش ساخت و راه اندازی ربات از ابتدا تا انتها از آدرس www.asemanarm.ir قابل دانلود می باشد و هر فرد علاقه مند به ساخت این ربات می تواند با دنبال کردن این آموزش و البته تلاش مستمر، این ربات و یا قسمتی از آن را برای خود بسازد.

با توجه به مواد اولیه مصرفی و تجهیزات مورد استفاده در ربات هزینه ساخت این ربات به حداقل ممکن کاهش پیدا کرده است. آموزش ارائه شده برای ساخت و راه اندازی نیازمند حداقل اطلاعات در مورد نرم افزار های استفاده شده در هر بخش طراحی می باشد، منابع مورد نیاز برای آشنایی یا فراگیری این نرم افزار ها در هر بخش ارائه خواهد شد. کار طراحی این ربات با بحث در مورد نحوه کارکرد و درجات آزادی این ربات شروع شد، در ادامه با توجه به عملکرد مورد نیاز، عملگر های ربات که شامل موتور های استفاده شده برای مفاصل بود، انتخاب شد. همچنین مواد اولیه لازم برای ساخت قسمت های مختلف مکانیک ربات مشخص شدند. در مرحله بعد، طراحی سه بعدی با توجه به ابعاد و ویژگی های ساختاری عملگر ها و مواد خام مورد استفاده انجام شد و نهایتا قطعات مکانیکی ربات با استفاده از مدل های سه بعدی ساخته و مونتاژ شدند. بعد از آماده سازی مکانیک ربات به طور کامل، مدار های الکترونیکی ربات طراحی و ساخته شد. با فراهم شدن بستر مناسب برای شروع تست های نهایی، برنامه های اولیه مربوط به تست حرکت موتور ها و سنسور ها به طور مجزا نوشته شد و تست های اولیه انجام شد. نهایتا بعد از انجام تست ها، برنامه اصلی ربات نوشته شد و مراحل مربوط به کالیبراسیون سنسور ها و تست های

عملکرد واقعی ربات انجام شد. مراحل ارائه شده در این آموزش دقیقا به ترتیب مراحل انجام شده برای ساخت ربات می باشد و به علاقه مندان توصیه می شود در صورت تمایل برای ساخت، همین ترتیب را در نظر داشته باشند. امید است تلاش شما منجر به ساخت نمونه های دیگر و حتی نمونه های توسعه یافته این ربات شود.

در حال حاضر گروه رباتیک آسمان در نظر دارد با جمع آوری سرمایه، ساخت نسخه بعدی این ربات با عملکرد واقعی تر و امکانات بیشتر را شروع کند. منابع ما برای ادامه کار بسیار محدود می باشد، بنابراین از همه علاقه مندان، فعالان در این حوزه و همه افرادی که به نحوی تمایل دارند به ما در ادامه این کار کمک کنند دعوت می شود سهمی هر چند کوچک در ادامه این پروژه داشته باشند. در صورت تمایل برای کمک به ما از گزینه "حمایت از ما" در سایت www.asemanarm.ir استفاده کنید. همواره قدردان حمایت های شما خواهیم بود.

۱. مکانیک ربات

۱,۱. شرح عملکرد و بررسی کلی مکانیک بازو و دست

مکانیزم اولیه طراحی شده یک بازوی ۱۰ درجه آزادی بود که به دلیل برخی محدودیت های موجود در پروژه به یک بازوی ۹ درجه آزادی تغییر داده شد. ۹ درجه آزادی شامل دو درجه آزادی در محل اتصال بازو به بدن (که عملکردی شبیه به مفصل گوی و کاسه در بدن انسان را شبیه سازی می کند)، دو درجه آزادی در آرنج (شامل خم شدن آرنج و حرکت پیچشی مچ دست) و پنج درجه آزادی در دست (شامل حرکت انگشتان دست به صورت کاملا مجزا) می باشد. حرکات مربوط به خم شدن مچ دست در این مدل وجود ندارد، که البته نحوه اضافه کردن این حرکات در آموزش مطرح شده است. در طول آموزش و از این قسمت به بعد زمانی که کلمه "بازو" به کار رفته است، منظور از کتف تا ناحیه مچ دست و زمانی که کلمه "دست" به کار رفته، منظور کف دست و پنج انگشت دست می باشد.

۱,۲. انتخاب مواد اولیه و عملگر های ربات

برای طراحی بازو ماده اولیه تخته سه لا (با ضخامت 3 mm) و روش ساخت برش لیزر در نظر گرفته شد. از دلایل انتخاب تخته سه لا به عنوان ماده اولیه می توان به قیمت مناسب و همچنین سبک بودن آن اشاره کرد. قابل ذکر است که سبک شدن بازو تا حد زیادی در کاهش هزینه های مربوط به عملگر ها و عملکرد مناسب تر بازو موثر است.

نمونه چوب استفاده شده برای طراحی

در طراحی بازو در قسمت های خاصی نیاز به مواد با استحکام بالاتر از چوب وجود داشت که از فایبرگلاس با ضخامت 2 mm استفاده شده است.

برای طراحی دست با توجه به حساس تر بودن عملکرد و شکل ظاهری این قسمت از PLA[1] (نوعی پلیمر که برای ساخت محصولات پلاستیکی مورد استفاده قرار می گیرد) به عنوان ماده اولیه و پرینت سه بعدی به عنوان روش ساخت انتخاب شد.

از سروو موتور ها به عنوان عملگر های مناسب برای بازو و دست استفاده شد. سه عامل گشتاور مورد نیاز، موجود بودن در بازار داخلی و قیمت برای انتخاب سروو موتور ها مورد توجه قرار گرفت و نهایتا مدل TowerPro MG995 برای بازو و TowerPro MG90s برای دست انتخاب شدند.

این سروو موتور ها را می توانید از لینک های زیر یا هر فروشگاه اینترنتی دیگر تهیه کنید:

http://shop.aftabrayaneh.com/Motors_Drivers/Servo_Motors/MG995_Metal_Gear_Servo.html

http://shop.aftabrayaneh.com/Motors_Drivers/Servo_Motors/Metal_Gear_Servo_MG90S.html

[1] http://en.wikipedia.org/wiki/Polylactic_acid

برای ساخت ربات به ۱۲ عدد سروو MG995 با گشتاور اسمی 13 kg.cm و 5 عدد سروو MG90s با گشتاور اسمی 1,8 kg.cmنیاز دارید. در صورت تمایل به اضافه کردن درجه آزادی دهم باید از مدل MG995 یک عدد اضافه سفارش دهید. در عمل گشتاور واقعی از گشتاور اسمی کمتر می باشد و با توجه به تست های انجام شده گشتاور در حالت پایدار برای موتور های بزرگ تر 10 kg.cm در نظر گرفته شد. ۵ سروو موتور کوچک برای کنترل حرکت انگشتان دست به کار می روند. از ۱۲ سروو موتور بزرگ ۴ عدد به طور موازی برای چرخش بازو از ناحیه اتصال به بدن و ۴ عدد برای بلند کردن بازو از همان ناحیه مورد استفاده قرار می گیرند. در محل آرنج نیز ۲ سروو موتور کار خم کردن آرنج و ۲ سروو موتور کار پیچش مچ را انجام می دهند. دلیل استفاده از سروو موتور ها به طور موازی تامین گشتاور مورد نیاز در هر یک از مفاصل است. سروو موتور هایی با گشتاور بالاتر نیز در بازار موجود می باشد که با توجه به تفاوت قسمت بالا با سروو های مورد نظر استفاده از سروو ها به طور موازی مورد توجه قرار گرفت.

۱,۳. طراحی سه بعدی و ساخت بازو

نرم افزار مورد استفاده برای مدل سازی سه بعدی SolidWorks می باشد. توجه کنید که برای مشاهده فایل های سه بعدی موجود بر روی سایت نیاز به نسخه 2015 یا بالاتر از این نرم افزار دارید و فایل ها با نسخه های قدیمی تر قابل مشاهده نمی باشند.

این نرم افزار از لینک زیر قابل دانلود است:

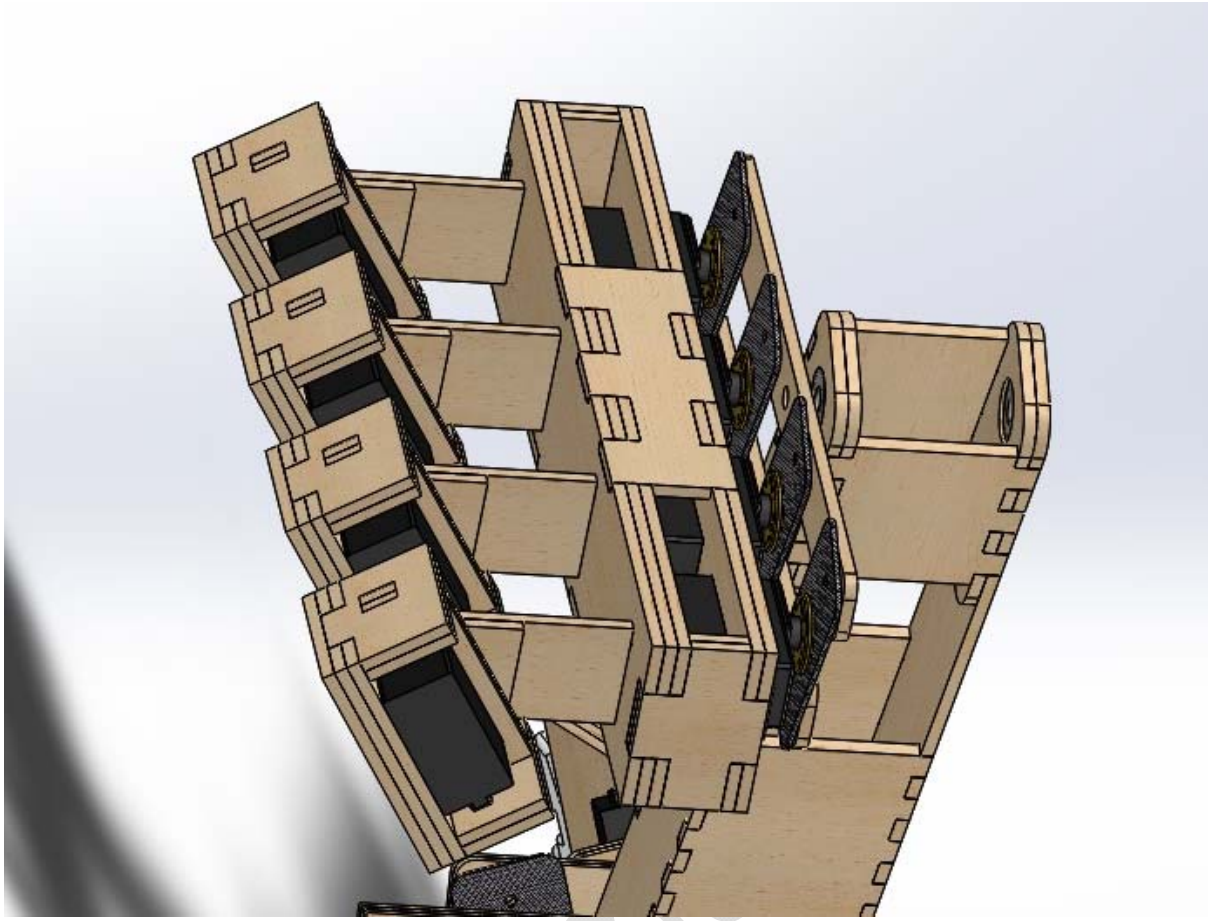
http://p30download.com/fa/entry/55086/

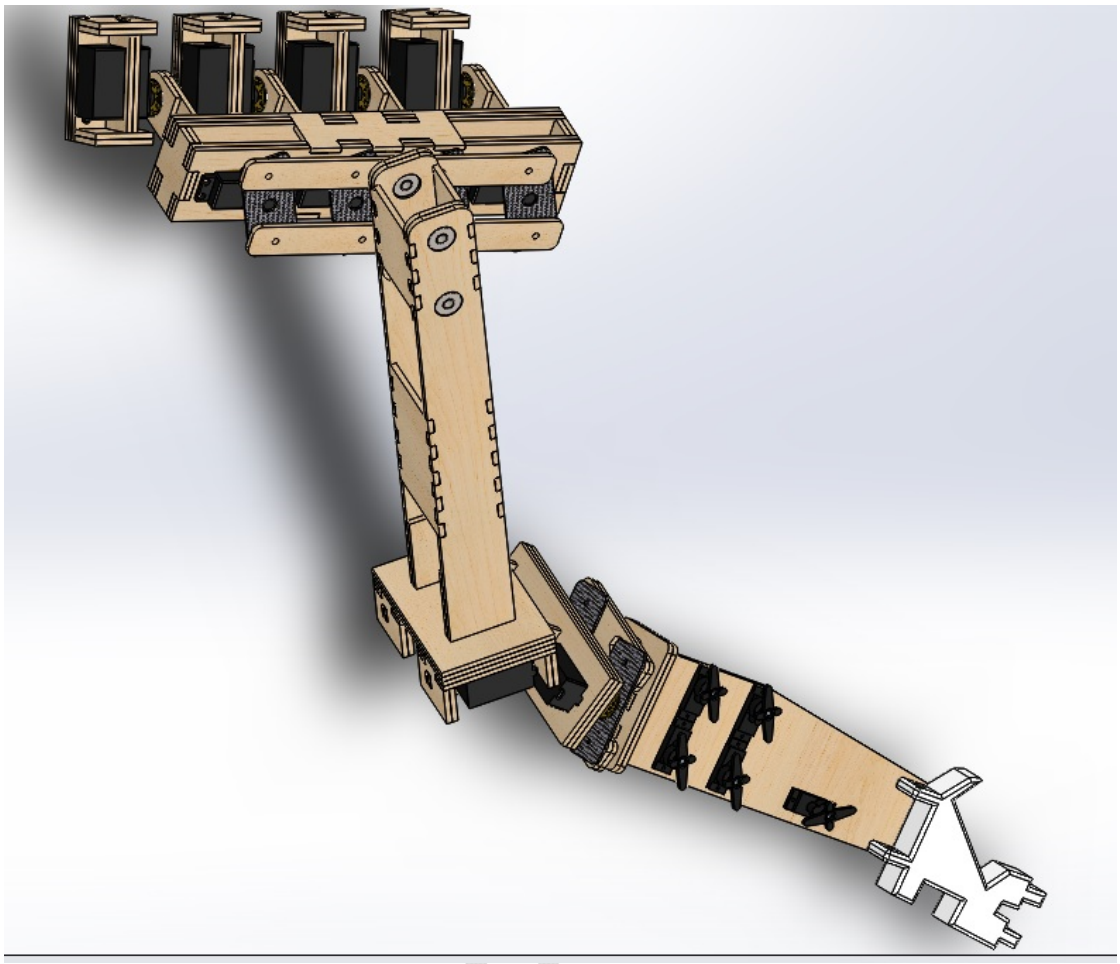
همچنین آموزش طراحی با این نرم افزار را می توانید از لینک زیر دانلود کنید:

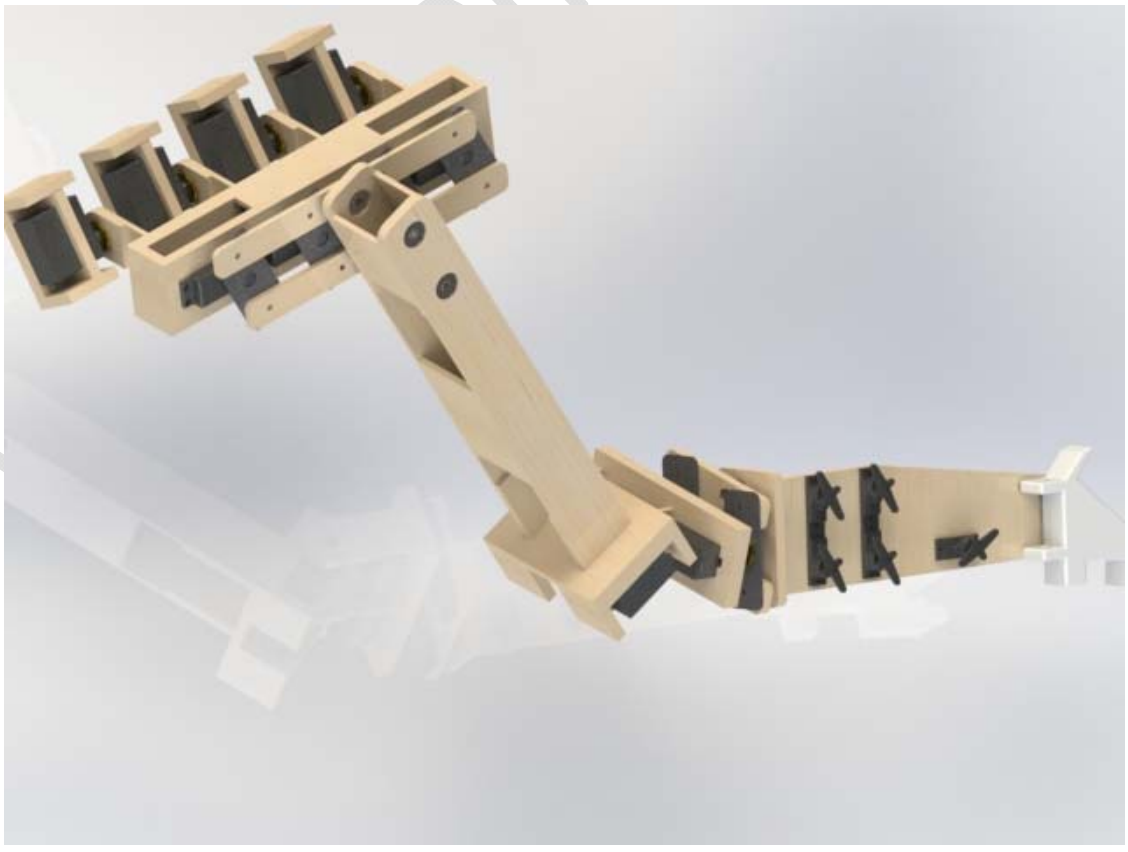
http://p30download.com/fa/entry/48733/

طراحی دست با توجه به ضخامت 3 mm چوب و اندازه های دقیق موتور ها انجام شده است. فایل های مربوط به مدل سه بعدی بازو را می توانید از قسمت "دانلود ها"ی سایت www.asemanarm.ir دانلود کنید.

در ادامه تصاویری از مدل های سه بعدی طراحی شده را مشاهده می کنید:

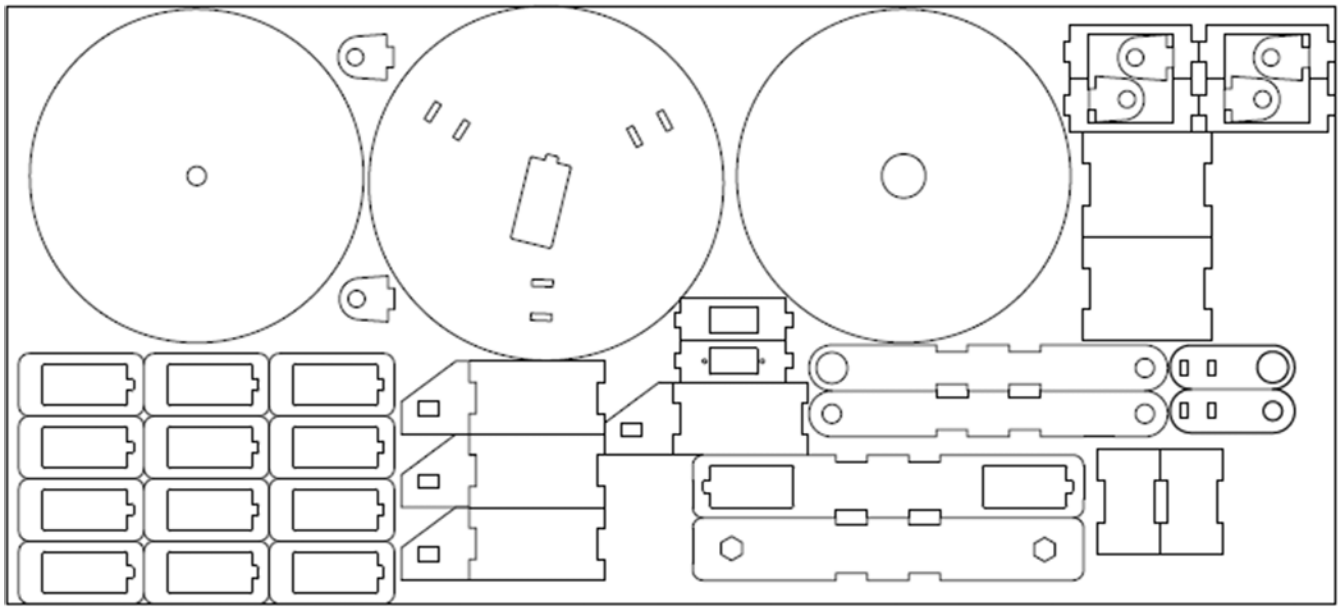




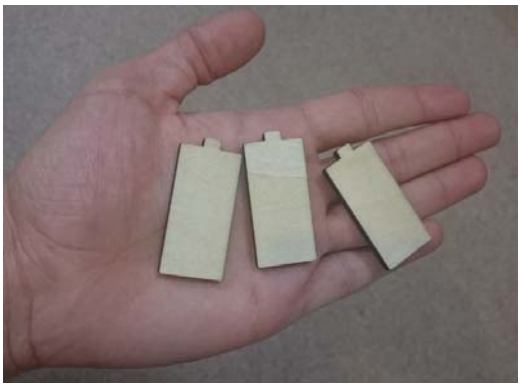


قسمت های طراحی شده توسط چوب و فایبرگلاس در مدل سه بعدی به طور کامل مشخص شده اند. توصیه می شود فایل ها را دانلود کنید و با دقت مورد بررسی قرار دهید.

پس از مدل سازی سه بعدی نوبت به ساخت بازو می رسد. در این مرحله لازم است تا فایل ها را برای برش توسط لیزر آماده کنید. برای این کار لازم است تا اشکال سه بعدی که به صورت قطعات با ضخامت 3 mm از چوب طراحی شده اند را به صورت تصاویر دو بعدی که در واقع خطوط برش لیزر هستند ذخیره کنید. برای این کار می توانید از نرم افزار هایی مانند AutoCAD یا CorelDraw استفاده کنید. برای ایجاد این فایل ها باید هر قطعه را به طور جداگانه با فرمت DWG ذخیره کرده و نهایتا تمامی قطعات را در یک فایل قرار دهید. شکل زیر نمونه ای از شکل مورد نیاز برای برش لیزر می باشد:

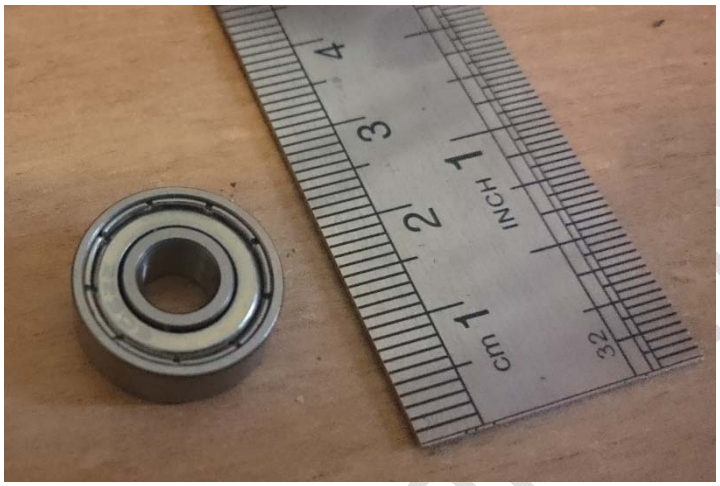


علاوه بر چوب فایبرگلاس با ضخامت کم نیز با لیزر قابل برش می باشد، برای این کا باید قدرت لیزر روی بیشترین مقدار قرار گرفته و چندبار خطوط مورد نظر را طی کند. قطعات بعد از برش به فرم زیر خواهند بود:



نمونه قطعات برش خورده با لیزر

علاوه بر قطعات برش داده شده از چوب و فایبر گلاس نیاز به تعدادی پیچ M6 و M3 نیز دارید. در محل مفاصل در مکانهایی که حرکت چرخشی موتور ها به حرکت رفت و برگشتی تبدیل می شود نیاز به بال برینگ های با قطر خارجی 17 mm و قطر داخلی 6 mm خواهید داشت. اگر به بال برینگ هایی با این مشخصات دسترسی نداشتید به راحتی می توانید در مدل سه بعدی قطر قسمت دایره ای از چوب را که بال برینگ در آن قرار گرفته تغییر دهید تا متناسب با بال برینگ های در دسترستان باشد.



تمامی اجزا مورد استفاده به جز پیچ ها در مدل سه بعدی مشخص شده اند، در مورد پیچ ها فقط سوراخ های عبور پیچ مشخص اند و با کمی دقت و اندازه گیری قطر سوراخ ها در مدل می توانید طول و قطر پیچ های مورد نظر را به دست آورید. برای متصل کردن چوب ها به هم نیز از چسب ۱ ۲ ۳ استفاده شده است. در قطعاتی که ضخامت بیشتر از یک لایه چوب دارند عملیات چسباندن لایه های مختلف روی هم باید با دقت بیشتری انجام شود، زیرا اندازه ها کاملا منطبق با قطعات مورد استفاده هستند و در صورت اشتباه ممکن است قطعات در جای خود جا نروند، در این قبیل از موارد می توانید از یک سوهان ریز برای جبران خطاها و منطبق کردن دوباره اندازه ها استفاده کنید.

تجهیزات جانبی سروو موتور ها شامل تعدادی سر سروو می باشد که اشکال مختلفی دارند.در بازو و برای سروو موتور های بزرگ از مدل دایره ای در طراحی و ساخت استفاده شده است.

برای اتصال این سر سروو ها به نقاط طراحی شده در بازو از پیچ های ریز که در تلفن های همراه کاربرد دارند استفاده شده است. به این صورت که سر سروو روی قطعه ای که باید به آن متصل شود قرار می گیرد، سپس در محل چند تا از سوراخ های سر سروو (حداقل ۳ تا)، سوراخ هایی روی قطعه توسط مینی دریل ایجاد می شود (قطر مته 1 mm) حال با یک پیچ گوشتی مناسب پیچ ها را داخل سوراخ ها سفت می کنیم. اگر پیچ با رزوه مناسب انتخاب شود، در هنگام پیچیدن شکل رزوه در قسمت پلاستیکی سر سروو و چوب ایجاد می شود و اتصالی با استحکام بالا پدید می آید. در صورت نیاز به باز و بسته کردن این پیچ ها باید دقت شود تا از به وجود آمدن مسیر های جدید برای رزوه ها جلوگیری شود.

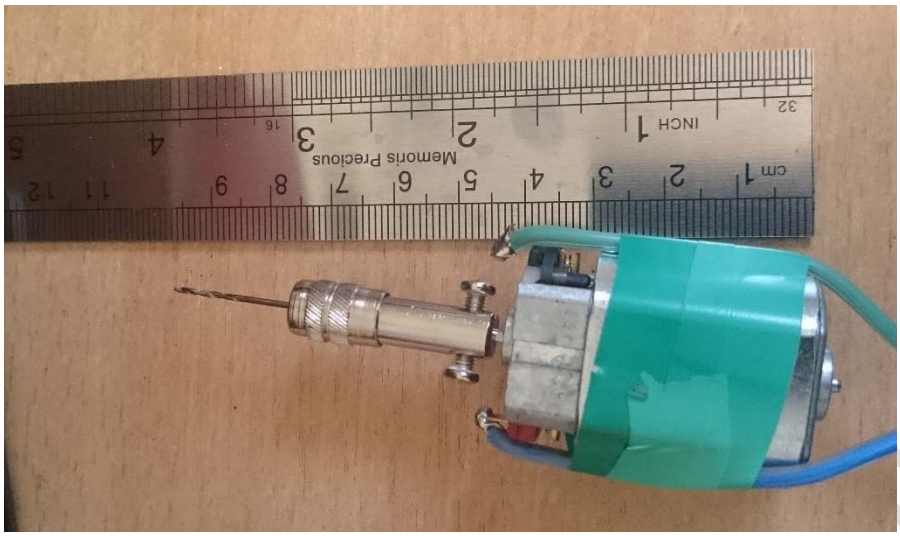
پیچ های مورد بحث را می توانید از فروشندگان قطعات یدکی موبایل یا برخی موبایل فروشی ها خریداری کنید.

این پیچ ها به صورت بسته ای شامل پیچ های مختلف فروخته می شوند

پیچ با رزوه نامناسب

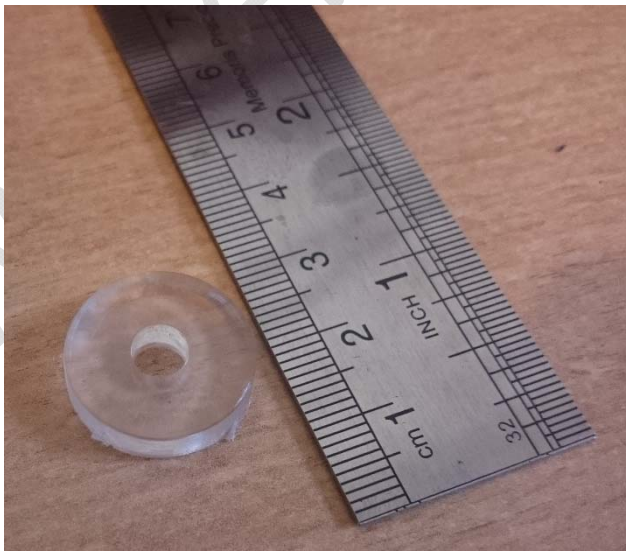
پیچ با رزوه مناسب

در صورتی که به مینی دریل دسترسی ندارید می توانید با استفاده از سه نظام و یک آرمیچر کوچک یک مینی دریل برای خود بسازید.



یکی از مشکلاتی که با موازی شدن سروو موتور ها به وجود می آید، عدم تطبیق موقعیت زاویه ای سروو موتور های موازی شده است. روش کار سروو موتور های معرفی شده به این صورت است که ۳ سیم دارند، که دو تا از این سیم ها برای تغذیه موتور به کار می روند و پالس دریافتی توسط سیم سوم موقعیت زاویه ای شافت سروو موتور را مشخص می کند، یک انکودر به طور مداوم موقعیت زاویه ای شافت را مورد بررسی قرار می دهد و این موقعیت را برای میکروکنترلر داخلی سروو موتور ارسال می کند. میکرو کنترل موقعیت دریافت شده توسط سیگنال خارجی و انکودر را با هم مقایسه کرده و در صورت تفاوت این دو عدد درایور موتور را در جهتی که موقعیت انکودر را با موقعیت ارسالی توسط پالس خارجی منطبق سازد راه اندازی می کند. بنابراین در صورت ثابت بودن سیگنال خارجی موتور همواره تمایل دارد در یک موقعیت ثابت قرار گیرد. در صورتی که بار خارجی به با مقدار نسبتا بالا به موتور اعمال شود و قصد تغییر موقعیت شافت را داشته باشد، به محض ایجاد تغییرات بسیار کوچک در موقعیت شافت موتور عکس العمل نشان داده و این تغییرات را به حالت قبل بر می گرداند. اثر مداوم بار و جبران شدن موقعیت توسط موتور معمولا به صورت لرزش در آن موقعیت خاص دیده می شود. زیاد بودن بار و تداوم این حالت رفت و برگشتی باعث جریان کشی بیش از حد موتور ها و داغ شدن آنها می شود. در برخی موارد نیز باعث سوختن موتور یا خرد شدن گیربکس در نمونه هایی با گیربکس پلاستیکی می شود. حال دو موتور را که به شکل موازی کار می کنند در نظر بگیرید. امکان به وجود آمدن چندین مشکل وجود دارد که برای هر کدام راه حل هایی ارائه شده است. مشکل اول مشخص نبودن موقعیت فعلی موتور هاست. به این معنی که موقعیت زاویه ای شافت هر عددی می تواند باشد. اگر در این حالت دو موتور موازی شوند، مثلا با دو بازو با طول یکسان به یک صفحه صلب متصل شوند و سپس موتور ها راه اندازی شده و پالس قرار گرفتن در موقعیت ۹۰ درجه برای هر دو ارسال شود، در حالتی که موقعیت اولیه هر دو موتور کاملا یکسان باشد به راحتی در موقعیت ۹۰ درجه قرار می گیرند ولی در صورتی که موقعیت اولیه موتور ها فرق داشته باشد

میزان چرخش دو موتور یکسان نخواهد بود و با توجه به صلب بودن اتصال حداقل یکی از موتور ها در موقعیت پالس ارسالی قرار نمی گیرد. این عامل باعث می شود تا هر یک از موتور ها مکانیزم را به سمت زاویه که برای آن موتور ۹۰ درجه تعریف شده ببرد و با توجه توضیحاتی که در مورد روش کارکرد سروو موتور داده شد نهایتا این تلاش زمانی پایان می یابد که یکی از موتور ها از بین رفته باشد. برای جلوگیری از این مشکل کافیست قبل از اتصال سر سروو ها به یک قسمت صلب آنها را توسط برد راه انداز و یک برنامه مناسب در موقعیت یکسان قرار داده و سپس اتصال را برقرار نمایید. مشکل بعد مربوط به زمانی است که به دلیل دقیق نبودن هر چند جزئی برخی از قطعات حتی با وجود یکسان بودن موقعیت اولیه موتور ها بعد از اتصال ۱ تا ۲ درجه اختلاف موقعیت بین موتور ها وجود خواهد داشت. یکی از راه های برطرف کردن این مشکل ارسال یک پالس جداگانه برای هر یک از موتور های موازی شده و کالیبره کردن زوایای ارسالی در برنامه است. به نحوی که این اختلاف زاویه جبران شود. این کار باعث زیاد شدن حجم برد و برنامه به مقدار زیاد خواهد شد و توصیه نمی شود بنابراین راه حلی مکانیکی برای این قسمت ارائه شده است. به این ترتیب که در محل اتصال سر سروو ها به قطعات سر سروو ها مستقیما به قطعه پیچ نمی شوند. ابتدا سر سروو ها به یک واشر لاستیکی پیچ شده و سپس ان واشر به چوب چسبانده می شود. این کار تا حد زیادی مشکلات به وجود آمده را برطرف می کند. محل قرار گیری واشر ها و نحوه اتصال آنها به خوبی در مدل سه بعدی مشخص شده است.



تصویر نمونه واشر

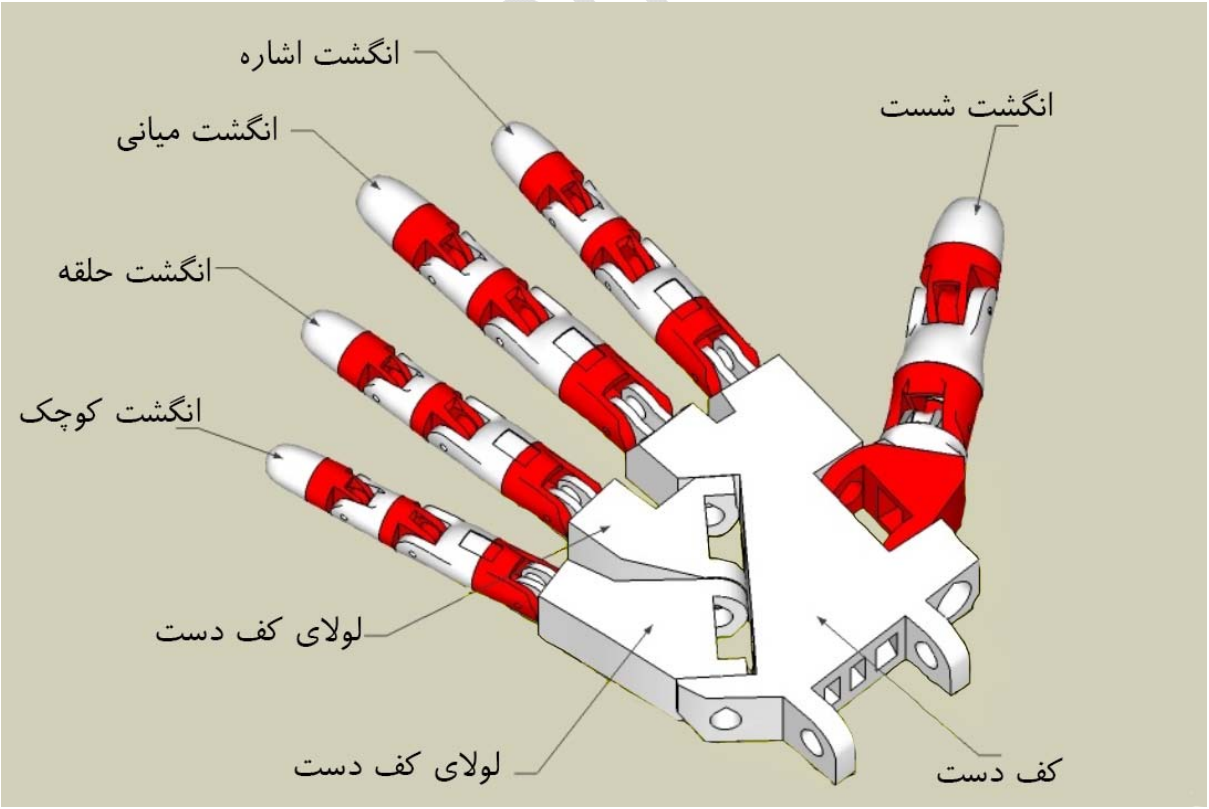
فایل های سه بعدی قابل دانلود شامل قطعات به صورت مجزا و مدل مونتاژ شده ربات می باشد که موقعیت تک تک اجزا را نسبت به هم به وضوح نشان می دهد و به راحتی می توان از آن برای سرهم کردن ربات راهنمایی گرفت.

توصیه می شود قبل از اقدام برای ساخت، یک بار این آموزش را تا انتها مرور کنید. زیرا بعضی اتصالات مکانیکی دائم را تا مراحل آخر نباید به هم متصل کنید.

۱,۴. طراحی سه بعدی و ساخت دست

پس از مدل سازی این قسمت فایل ها با فرمت stl ذخیره شده تا به عنوان ورودی برای نرم افزار های چاپ سه بعدی مرسوم مورد استفاده قرار گیرد. بنابراین امکان ادیت این فایل ها با نرم افزار Solidworks وجود ندارد. ولی در صورت تمایل با تغییر سایز کلی اجزا می توانید دست در اندازه های مختلف را بسازید. سایز فعلی دست تقریبا مشابه با یک انسان معمولی بالغ است. فایل های سه بعدی مربوط به دست به طور مجزا برای هر قسمت آماده شده اند. فایل مربوط به تک تک انگشتان، کف دست و ... به صورت جداگانه درون یک فایل زیپ روی سایت قرار گرفته و قابل دانلود است.

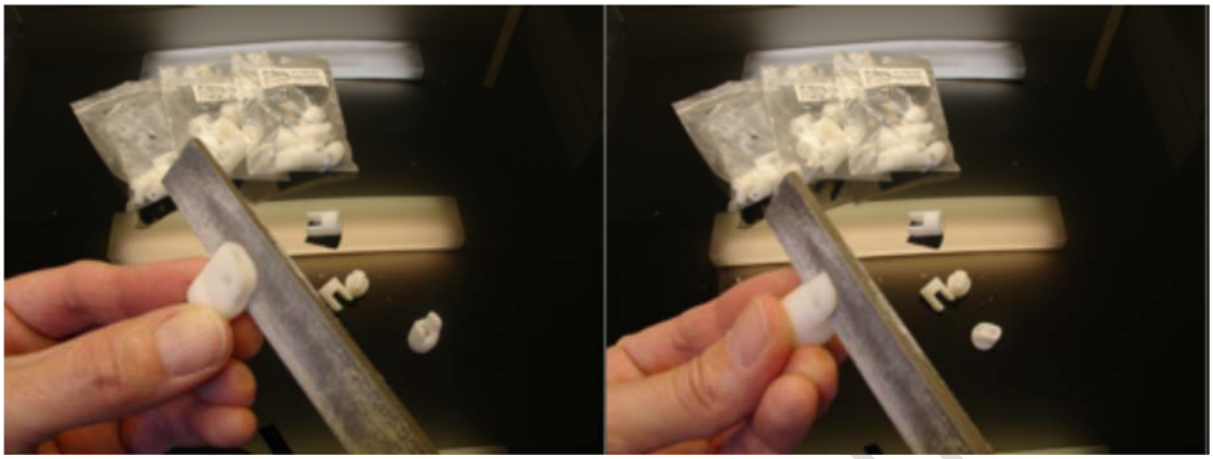
با استفاده از شکل زیر موقعیت هر مجموعه از قطعات پرینت گرفته مشخص می شود. سعی کنید پس از پرینت گرفتن فایل مربوط به انگشتان دست هر یک را در بسته ای جداگانه نگهداری کنید تا تشابه میان قطعات باعث سردرگم شدن شما نشود. فایل ها قابل دانلود همنام با نوشته های روی شکل هستند.



بعد از پرینت قطعات دست باید سوراخ های مربوط به اتصالات را با مته مناسب به اندازه مورد نیاز رساند (فرایند چاپ سه بعدی استفاده شده دقت کافی برای سوراخ ها را ندارد).

توجه کنید که قطعات در محل لولا شدن به قطعه بعد سوراخ کاری می شوند. برای قطعات خارجی از مته 3 mm و برای قطعات داخلی از مته 3.2 mm یا 3.5 mm استفاده کنید.

شاید نیاز باشد قطعات را برای این که به شکل مناسب و روان در کنار هم حرکت کنند کمی سوهان بکشید(با توجه به دقت چاپ سه بعدی میزان سوهان کشیدن متفاوت است و در حالت بد بعضی تا ۱,۵mm نیاز به سوهان کشیده شدن دارند).

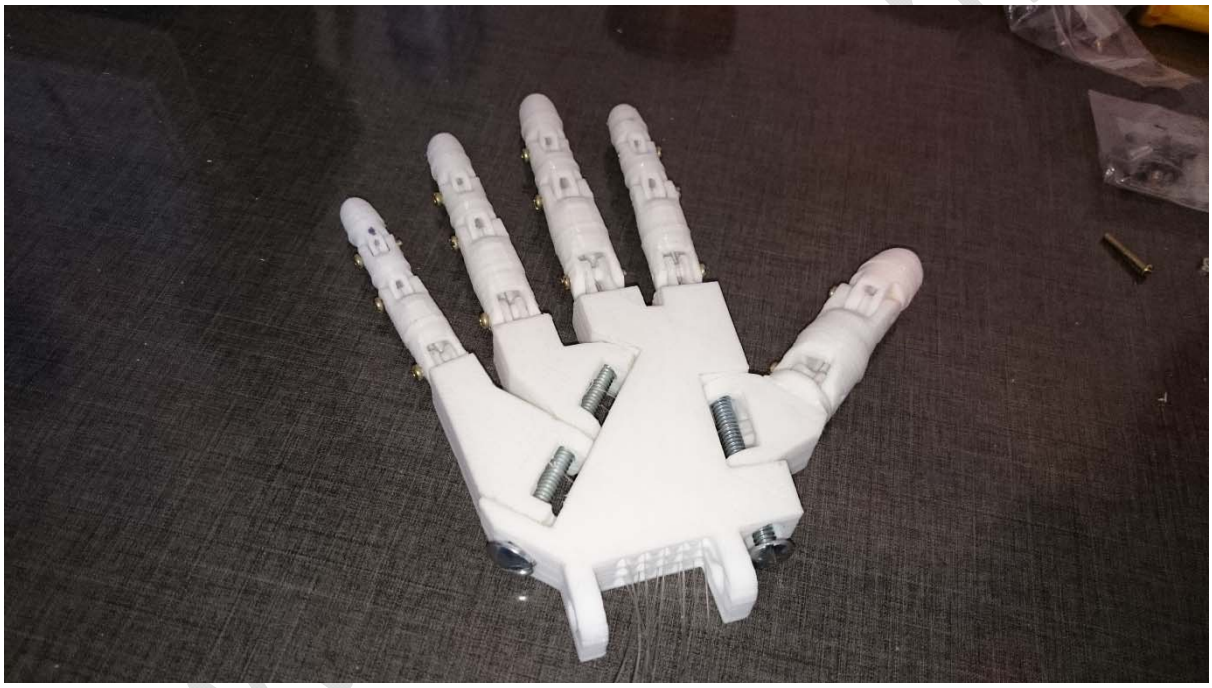


سپس لازم است با استفاده از چسب ۱ ۲ ۳ قطعات انگشت ها را از محل هایی که لولا نمی شوند به هم بچسبانید. قسمت سر انگشت ها را نگه دارید تا پس از عبور نخ آن را بچسبانید.

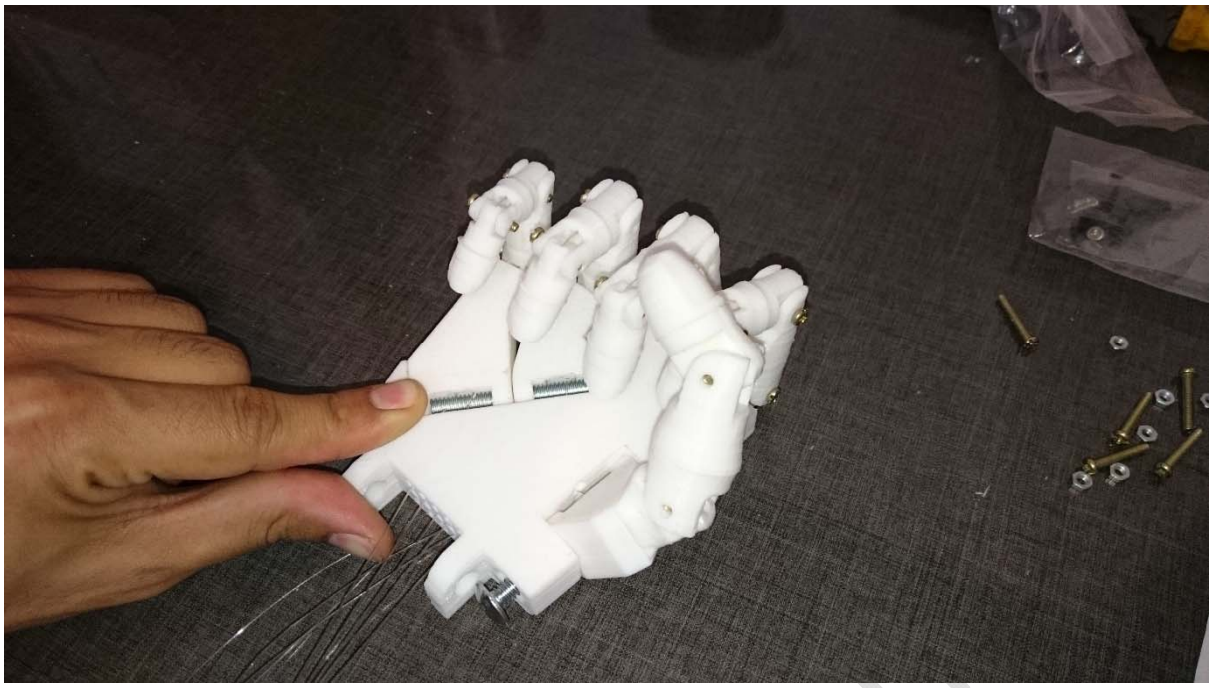
پس از انجام سوراخ کاری، سوهان کاری و آماده سازی انگشت ها قطعات دست را کنار هم بچینید.

حال باید از محل مفاصل انگشت ها پیچ M3 و مفاصل کف دست پیچ M8 عبور دهید. پس از آن نوبت به عبور نخ ها می رسد. از هر انگشت ۲ نخ عبور می کند. یک نخ از روی انگشت و دیگری از پشت

انگشت. در صورتی که نخ جلوی انگشت کشیده شود و نخ پشت انگشت آزاد شود انگشت خم می شود و برعکس همین عمل برای باز کردن انگشت به کار می رود. بعد از عبور این نخ ها از داخل انگشت و خارج کردن آنها از سر انگشت، نخ ها را در همان نقطه گره بزنید. روی سر انگشت ۲ سوراخ وجود دارد که یکی برای نخ روی انگشت و دیگری برای نخ پشت انگشت می باشد. حال سر دیگر نخ را از داخل شیار های طراحی شده درون کف دست عبور دهید تا انتهای نخ از محل مچ خارج شود. برای هر انگشت این عمل را تکرار کنید. درون دست به ازای هر انگشت ۳ سوراخ وجود دارد. از سوراخ های بالا و پایین برای عبور دو نخ مورد نیاز برای جمع و باز کردن انگشت استفاده می شود. سوراخ میانی در این طرح کاربردی ندارد ولی برای عبور سیم سنسور در صورت وجود سنسور لامسه مورد استفاده قرار می گیرد. پس از انجام فرایند های بالا دستی به شکل زیر خواهیم داشت.

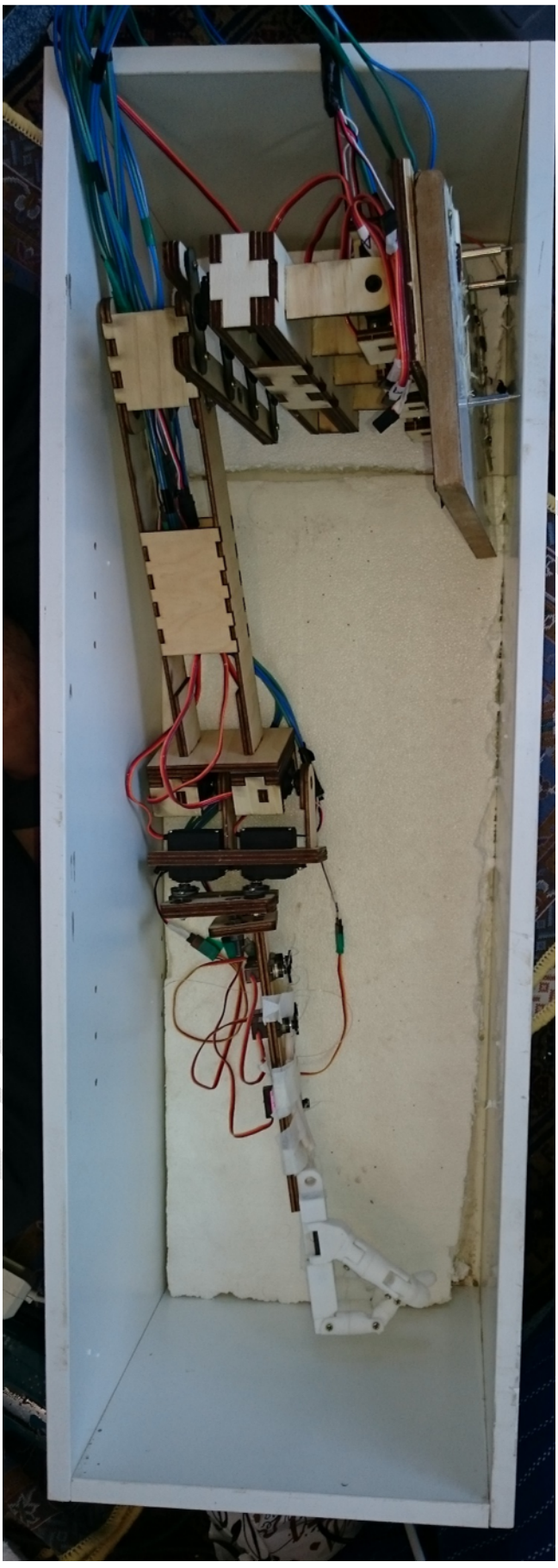


حال با کشیدن نخ ها باید قادر باشید تا انگشتان دست را جمع و باز کنید، اگر موفق نشدید مسیر های عبور نخ را بررسی کنید.



بعد از آن سروو موتور های کوچک به نخ ها متصل می شوند تا وظیفه جمع و باز کردن انگشتان را بر عهده بگیرند. محل نصب این سروو موتور ها در قسمت طراحی بازو در نظر گرفته شده است. کافی است یک سر سروو مناسب مانند شکل زیر انتخاب کنید و دو نخ مربوط به یک انگشت را به دو انتهای آن متصل کنید. بهتر است یکی از حالت های مرزی (انگشت کاملا باز یا انگشت کاملا بسته) را در نظر گرفته و در همان حالت نخ ها را در کشیده ترین حات ممکن به سر سروو ها متصل کنید تا با جزئی ترین حرکات سر سروو ها انگشت مورد نظر در موقعیت مناسب قرار گیرد. نخی که برای این کار استفاده می کنید باید تا حد ممکن محکم باشد و کش نیاید. نخ ماهیگیری انتخاب مناسبی برای این کار می باشد که می توانید از فروشگاه های لوازم شکار تهیه کنید.

نهایتا بعد از اسمبل کردن دست و بازو مکانیزم زیر را خواهیم داشت.



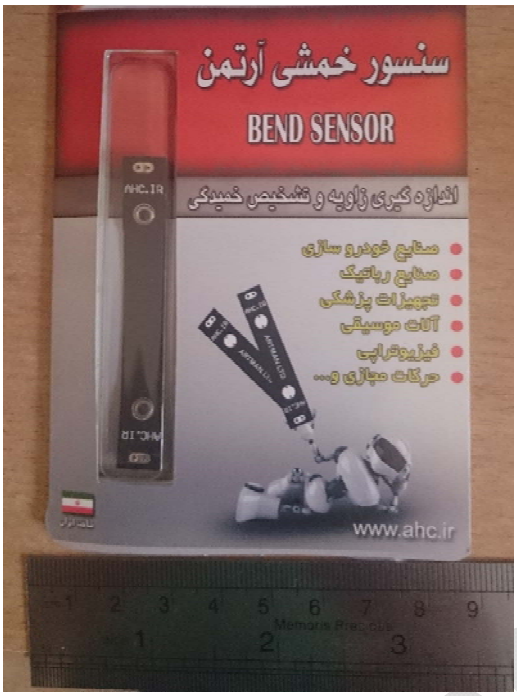
۲. الکترونیک ربات

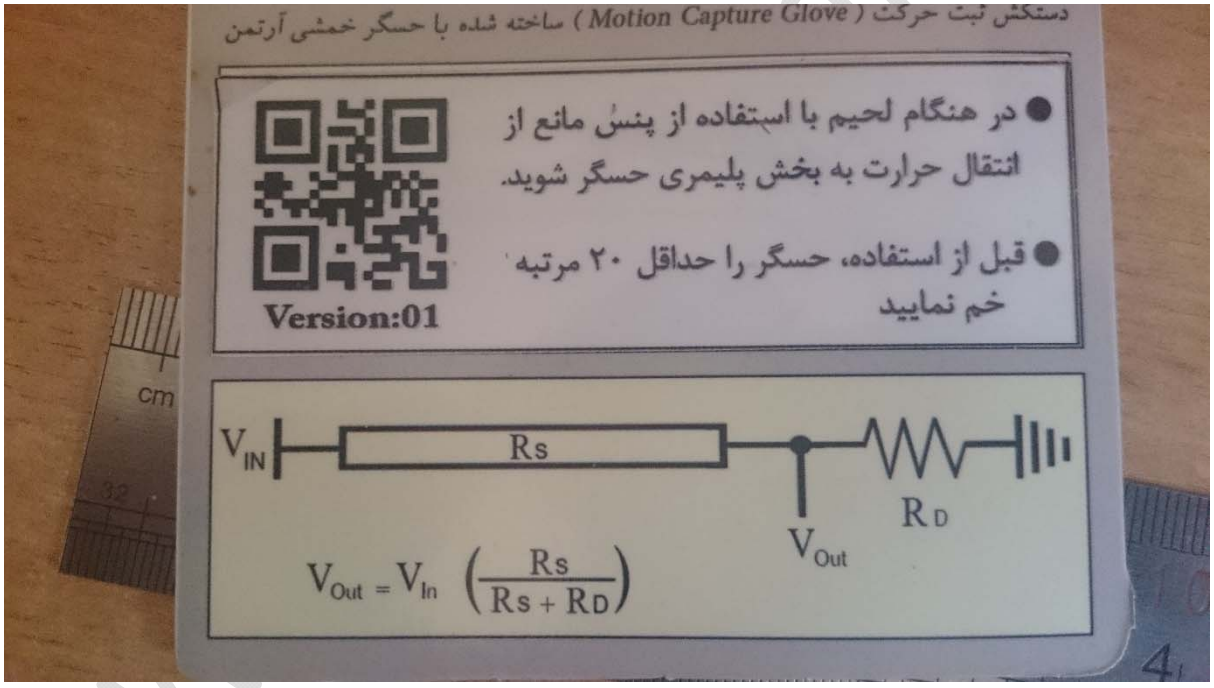
۲,۱. برآورد نیاز ها

در این قسمت به بردی نیاز داریم تا اطلاعات سنسور های مربوط به هر درجه آزادی را از دست واقعی انسان دریافت کرده و پس از پردازش، موقعیت هر سروو موتور را به صورت سیگنال مناسب برای آن ارسال کند. همچنین نوع سنسور ها و مدار مورد نیاز برای راه اندازی آنها نیز باید در نظر گرفته شود. خوشبختانه موتور های مورد استفاده دارای درایور داخلی هستند و نیازی به طراحی درایور به طور مجزا برای آنها وجود ندارد. با توجه به این نکته که از ۱۷ عدد موتور استفاده شده که در شرایط خاص امکان دارد همه هم زمان شروع به حرکت کنند یک منبع تغذیه سوئیچینگ ۵ ولت و ۳۰ آمپر برای تغذیه ربات در نظر گرفته شد. در ادامه به بررسی هر یک از موارد فوق می پردازیم.

تصویری از منبع تغذیه ربات

سنسور های مورد استفاده در ربات سنسور های خمشی هستند. این سنسور ها در واقع مقاومت متغیر هستند. تغییر در مقاومت آنها متناسب با میزان خم شدنشان می باشد. نمونه های زیادی از این سنسور ها وجود دارد که در اینجا از سنسور آرتمن که از تنونه های ایرانی این سنسور می باشد استفاده شده است. با یک جستجوی ساده در اینترنت می توانید فروشگاه مناسب برای خرید این سنسور را پیدا کنید.





مدار تقسیم ولتاژ پیشنهادی شرکت سازنده

برای قسمت پردازش ربات از برد های Arduino استفاده شده است. این برد ها معمولا شامل یکی از میکروکنترلر های خانواده AVR هستند که به همراه امکانات مورد نیاز برای راه اندازی و همچنین پروگرمر، همه در قالب یک برد ارائه می شوند. این برد ها به سادگی و از طریق پورت USB به کامپیوتر متصل شده و قابل برنامه نویسی می باشند. شرکت سازنده یک IDE اختصاصی نیز دارد که از زبان C پشتیبانی می کند. اپن سورس بودن و وجود کتابخانه های قدرتمند برای این برد باعث شده تا به طور گسترده ای در سطح جهان مورد استفاده قرار بگیرد. برای آشنایی با انواع برد ها، فراگیری زبان برنامه نویسی این برد

ها و همچنین دانلود IDE آن می توانی به سایت www.arduino.cc مراجعه کنید. برد های ارائه شده توسط این شرکت تنوع بسیار زیادی دارند و با توجه به نیاز می توان نمونه مناسب را انتخاب کرد. با توجه به تعداد بالای ورودی های آنالوگ مورد نیاز برای دریافت اطلاعات سنسور ها برد MEGA انتخاب شد. این برد از آدرس زیر با قیمت مناسب قابل تهیه کردن است:

http://shop.aftabrayaneh.com/Arduino_Boards/Arduino_Mega2560_R3.html

مشخصات کامل این برد ها از لینک بالا قابل دسترسی است.

۲,۲. طراحی نرم افزاری برد ربات

برای طراحی برد ربات از نرم افزار Altium Designer استفاده شده است. برای باز کردن فایل های مربوط به طراحی برد ربات به نسخه ۱۳ یا بالاتر از این نرم افزار نیاز خواهید داشت.

برای دانلود این نرم افزار می توانید به لینک زیر مراجعه کنید:

http://p30download.com/fa/entry/30844/

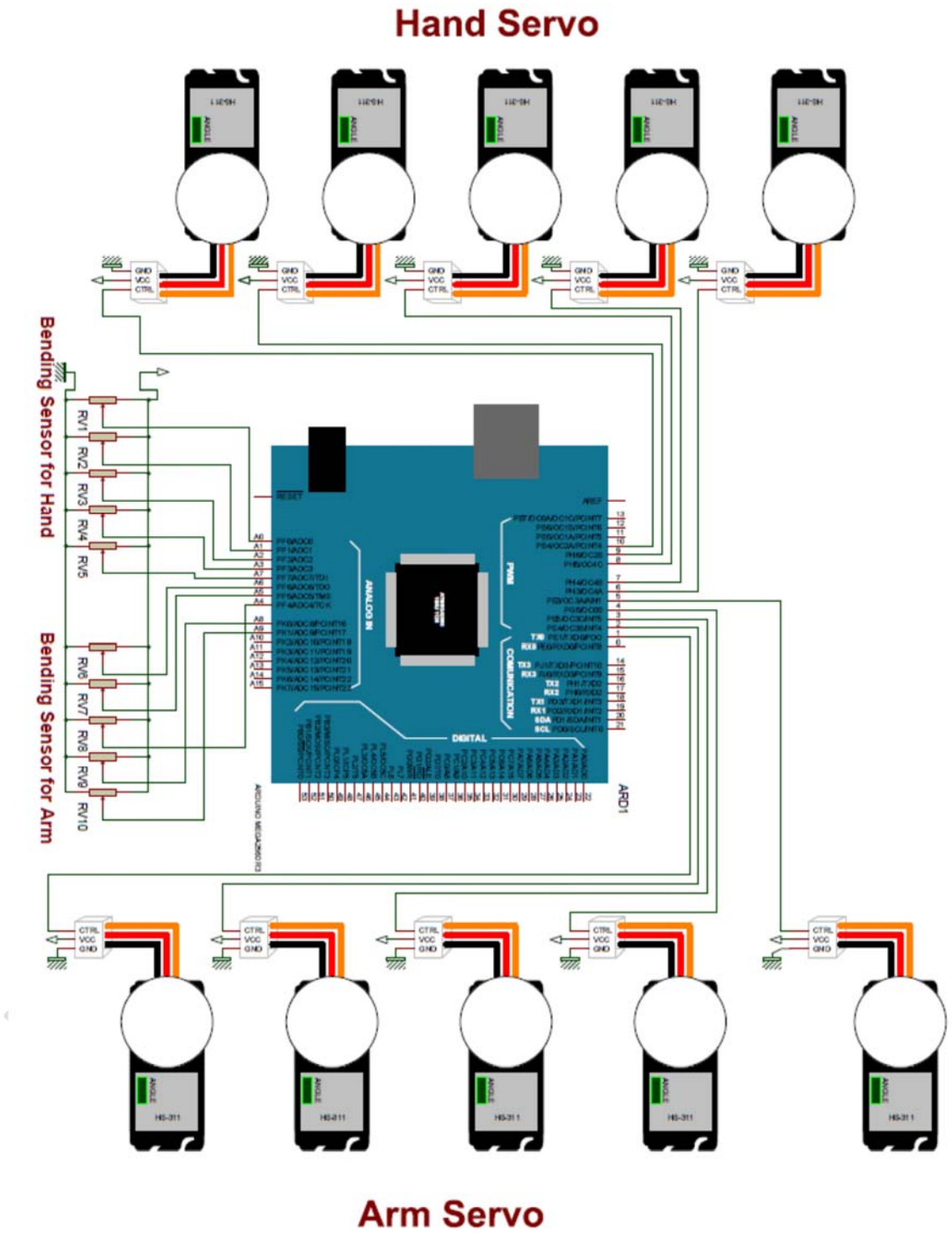
یا

http://p30download.com/fa/entry/55921/

برای فراگیری کار با نرم افزار می توانید از لینک زیر استفاده کنید:

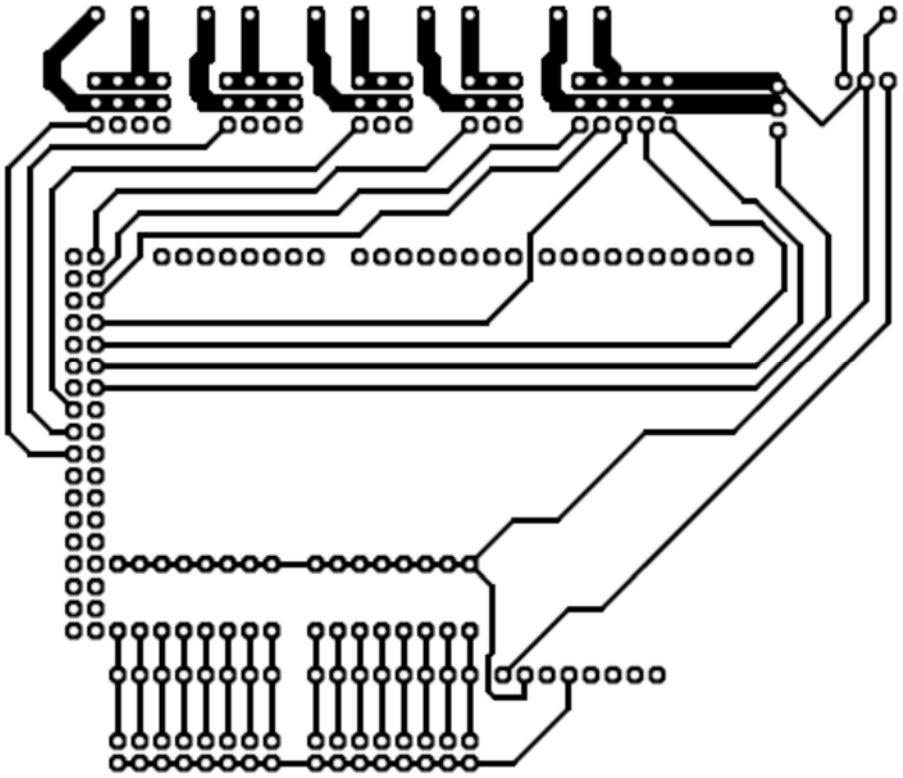
http://maktabkhooneh.org/course?course=protel980

شماتیک زیر کلیت موارد مورد نیاز برای برد را نشان می دهد. در این قسمت سنسور های خمشی با نماد مقاومت متغییر نمایش داده شده اند:



شکل بالا صرفا یک شمای کلی از مدار می باشد و شماره پین های ورودی و خروجی در مدار اصلی متفاوت بوده و با توجه به برنامه ربات مشخص می شود. طراحی مدار تقسیم ولتاژ سنسور ها طبق نمونه ارائه شده

توسط شرکت سازنده سنسور انجام شده و در آن مقاومت ثابت مقدارش 1MΩ در نظر گرفته شده است. پس از طراحی شکل مسیر های موجود روی برد مشخص می شود.



۲,۳. ساخت برد ربات

پس از آماده سازی الگوی بالا (این الگو در فایل قابل دانلود برای برد موجود است و در صورتی که تمایلی برای طراحی آن ندارید کافیست فایل PDF مربوطه را روی کاغذ A4 با سایز واقعی پرینت بگیرید) کافیست تا با یکی از روش های موجود آن را روی برد مسی پیاده سازی کنید و عملیات اسید کاری را روی آن انجام دهید. در ادامه یکی از روش های مرسوم برای پیاده سازی الگو روی برد توضیح داده می شود، روش های دیگر و همچنین نحوه اسید کاری را می توانید با یک جستجوی ساده در سطح اینترنت پیدا کنید.

روش اتو:

برای انتقال الگو روی برد به این روش به وسایل زیر نیاز خواهید داشت:

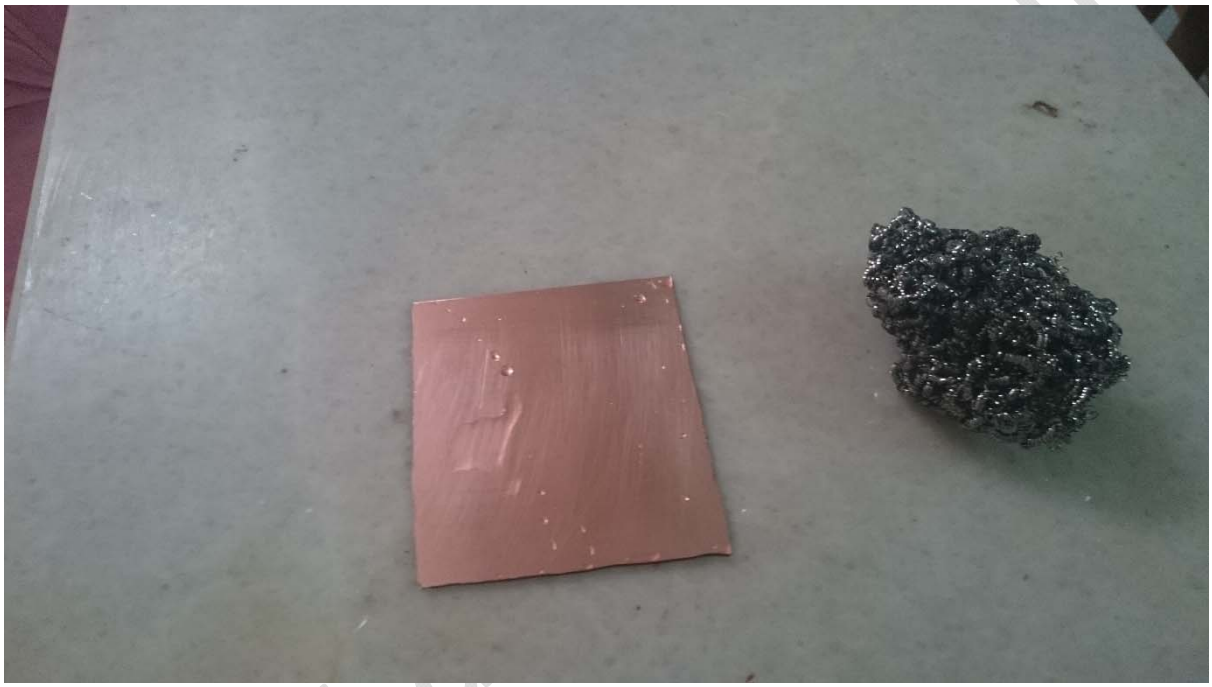
برد مسی

کاغذ گلاسه

پرینتر لیزری

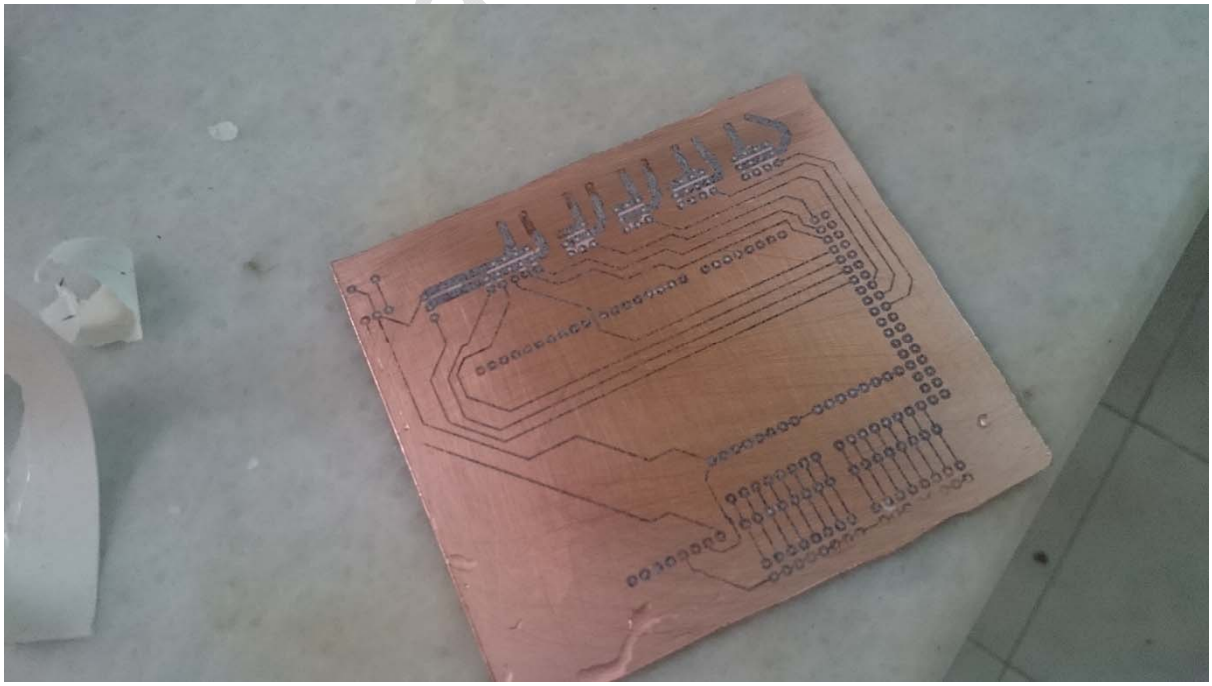
اتو

ابتدا الگو را در سایز واقعی روی کاغذ گلاسه توسط پرینتر لیزری و به صورت سیاه و سفید چاپ کنید. سپس برد مسی را با سمباده نرم یا سیم ظرفشویی بسابید تا لایه اکسید روی مس به طور کامل از بین برود. یکنواخت شدن سطح مس تاثیر بسیار خوبی در مراحل اسید کاری و لحیم کاری خواهد داشت.



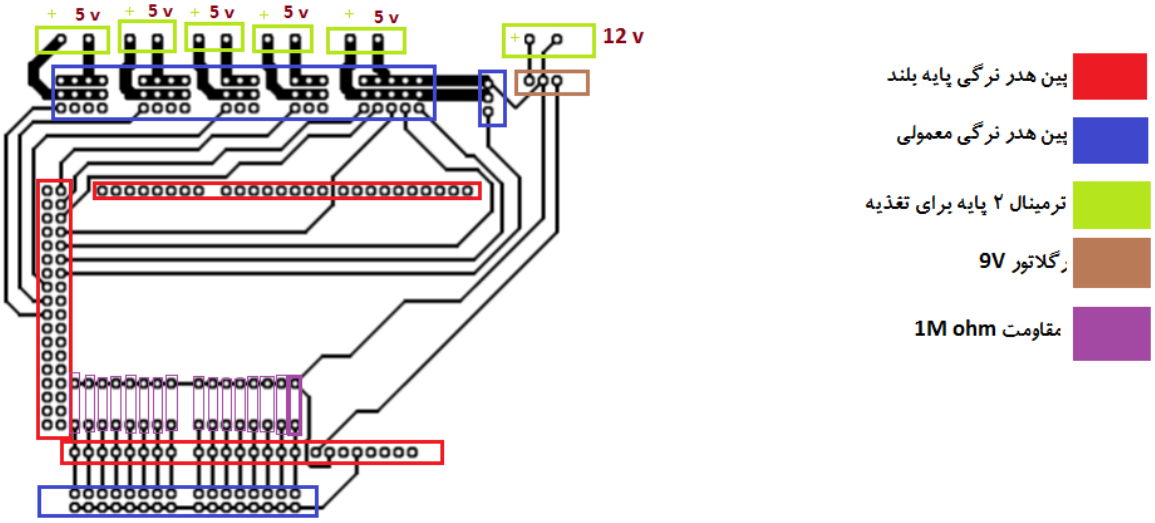
سپس کاغذ گلاسه که طرح مورد نظر را روی آن چاپ کرده اید از جهتی که الگو روی آن قرار دارد روی برد مسی قرار دهید.

حال درجه حرارت اتو را روی زیاد تنظیم و به مدت ۱۰ دقیقه برد و کاغذ روی آن را اتو کنید. مراقب لبه ها باشید این قسمت ها معمولا بیشتر دچار مشکل می شوند، قبل از انتهای کار این قسمت ها را دوباره اتو کنید. از اعمال حرارت بیش از حد به برد خودداری کنید زیرا باعث بخش شدن جوهر و خراب شدن الگو می شود. پس از مدت زمان گفته شده طرح به برد مسی انتقال یافته حال باید برد را زیر آب گرفته و قسمت های کاغذی را که به برد چسبیده اند جدا کنید. دقت کافی را در این مرحله به کار ببرید تا تمامی کاغذ هایی که نباید روی برد باقی بمانند را جدا کنید تا پس از اسید کاری برد شما کیفیت مطلوبی داشته باشد.

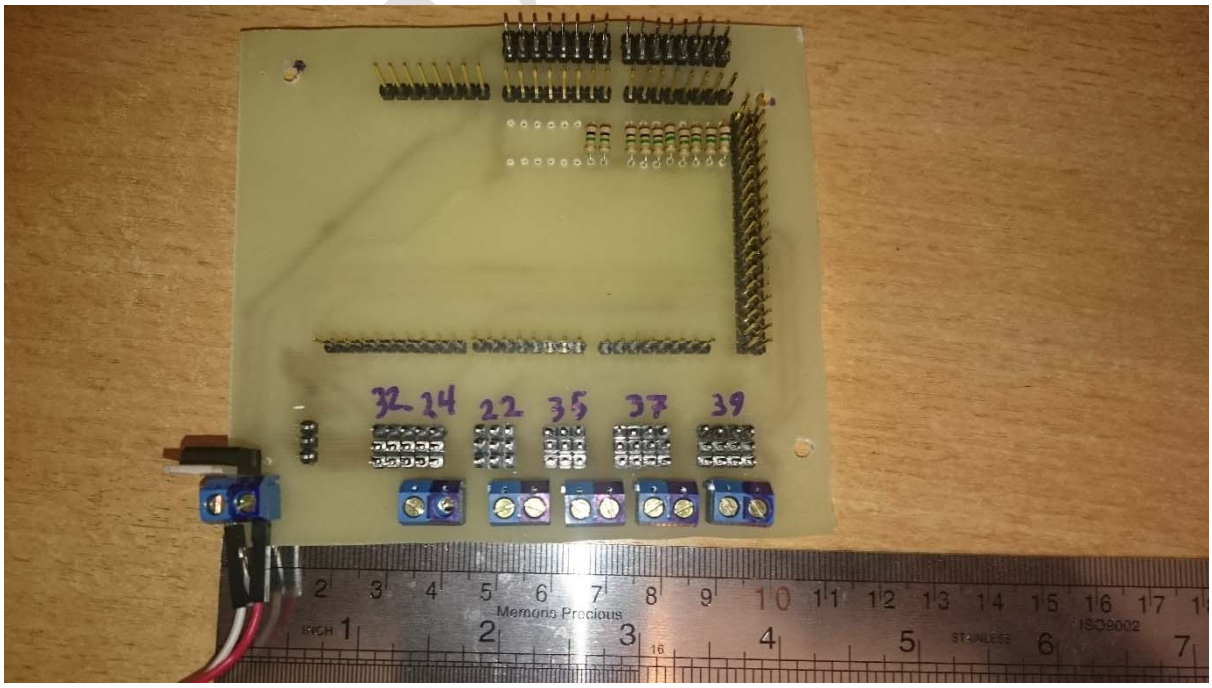


حال برد شما آماده اسید کاری است، بعد از اسید کاری نیاز است دوباره با سیم ظرفشویی اثر جوهر را از روی برد پاک کنید تا سطح مسی آن نمایات شود. سپس برد را با مینی دریل یا با آرمیچر و سه نظام (در بخش مکانیک توضیح داده شد) در محل های مناسب سوراخ کنید.

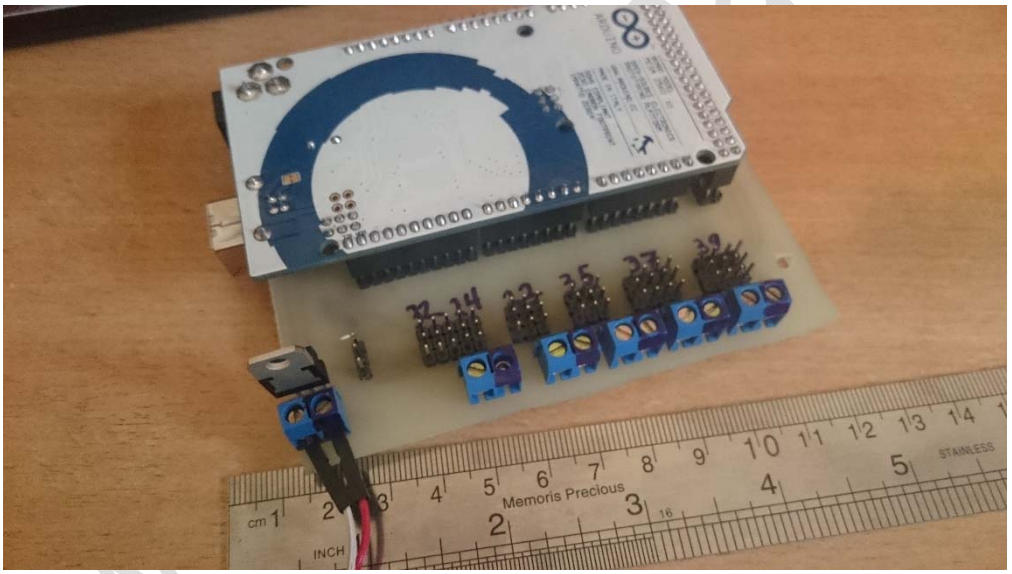
سپس مطابق با طرح زیر قطعات را روی برد لحیم کنید.

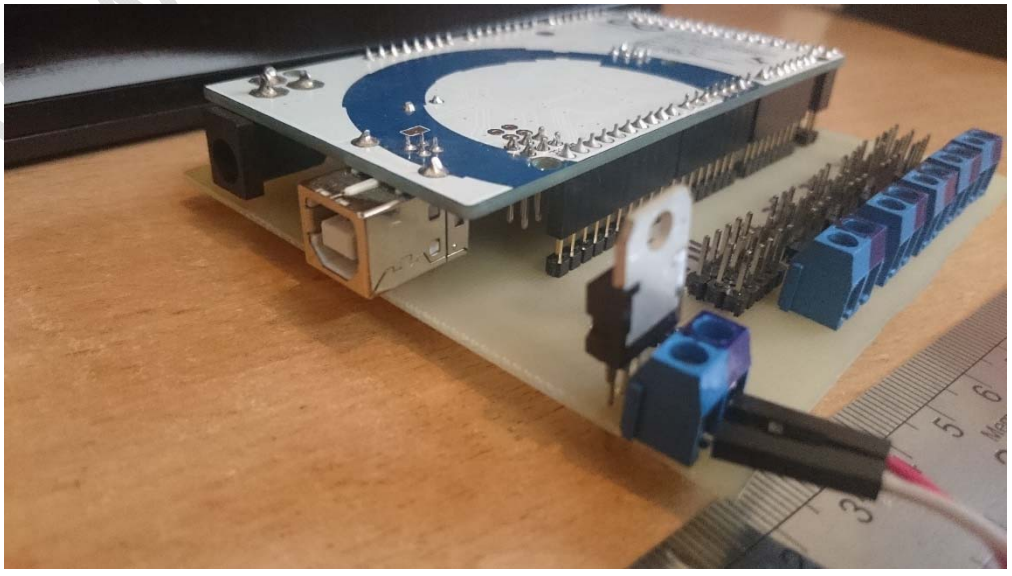


در نهایت بردی مطابق با شکل زیر خواهید داشت:



چهار پایه ای شماره ۳۹ بالای آن نوشته شده برای راه اندازی چهار موتور موازی شده یکی از مفاصل کتف و چهار پایه ای که عدد ۳۷ بالای آن نوشته شده برای راه اندازی مفصل دوم کتف می باشد(هر سروو موتور طبق توضیحاتی که قبلا در بخش مکانیک داده شده ۳ پایه دارد که به راحتی به برد بالا متصل می شود). سری پایه های مشخص شده با اعداد ۳۵ و ۲۲ برای راه اندازی موتور های مفاصل موجود در آرنج می باشند که در هر سری یک خروجی اضافی نیز در نظر گرفته شده تا در صورت نیاز موتور های بیشتر به مدار اضافه شود. پایه هایی که با شماره ۲۴ تا ۳۲ مشخص شده اند نیز به ترتیب برای راه اندازی سروو موتور های موجود در دست می باشند که هر یک سیگنال جداگانه دریافت می کنند. ترمینال آبی رنگ جلوی هر سری از پایه ها مربوط به تغذیه موتور های مرتبط به آن پایه می باشد. قسمت رنگی شده روی ترمینال ها نشان دهنده اتصال مثبت منبع تغذیه می باشد. پس از عبور از این مرحله نوبت به اتصال برد به پردازنده ربات می رسد. برد به نحوی طراحی شده که این امکان را به راحتی محیا می کند و مانند تصویر زیر می توانید پردازنده را روی برد قرار دهید.





۳. برنامه نویسی ربات

۳,۱. برآورد نیاز ها

در این قسمت نیاز به برنامه ای داریم که علاوه بر خواندن اطلاعات سنسور ها و اعمال داده های پردازش شده روی موتور ها در مراحل مختلف تست نیز قابل استفاده باشد. در قسمت بعد برنامه مورد نظر به طور کامل نوشته شده و توضیحات اضافی در قسمت های مورد نیاز بیان شده است. سورس کامل برنامه از قسمت دانلود های سایت www.asemanarm.ir قابل دانلود کردن است. برنامه نویسی در محیط برنامه نویسی Arduino انجام شده که کامپایلر مورد نظر از آدرس زیر قابل دانلود است.

http://www.arduino.cc/en/Main/Software

۳,۲. برنامه ربات و تست عملکرد

در این قسمت برنامه ربات را مورد بررسی قرار می دهیم:

```
#include <Servo.h>
```

کتاب خانه ای مناسب برای تولید سیگنال موقعیت سروو موتور ها

```
//Sensor vriables
```

تعریف متغیر هایی که داده های سنسور ها در آنها ذخیره می شود

```
int a;
int b;
int c;
int d;
int e;
int f;
int g;
int h;
int i;
```

```
//Servo variable
```

تعریف متغیر های مورد نیاز برای راه اندازی سروو موتور ها

```
Servo salman1;
Servo salman2;
Servo salman3;
Servo salman4;

Servo salman5;
Servo salman6;
Servo salman7;
Servo salman8;
Servo salman9;
```

```
//number of data for smoothing function
```

در این برنامه از یک فیلتر پایین گذر نرم افزاری برای کم کردن ارتعاشات تاثیر گذار روی سنسور ها استفاده شده است. این فیلتر تعدادی از داده ها را در طول زمان ذخیره کرده و میانگین آنها را به عنوان خروجی تحویل می دهد. برای مثال اگر سنسور در یک لحظه مشخص یک داده خاص را به عنوان ورودی به سیستم بدهد، داده خروجی برابر با میانگین داده ذکر شده و تعدادی از داده های قبلی خواهد بود. مقدار نسبت داده شده به متغیر زیر تعداد داده های ذخیره شده در این فیلتر را مشخص می کند. بدیهیست که با افزایش این عدد ارتعاشات سیستم کم شده ولیسرعت واکنش آن نیز به همان نسبت کاهش می یابد.

```
const int numReadings = 10;
```

```
//Arrays for needed for smoothing function
```

فیلتر یاد شده به صورت ۹ تابع هم شکل برای ۹ سنسور نوشته شده است. متغیر های زیر داده های گفته شده در قسمت قبل را برای هر یک از این توابع در خود ذخیره می کنند.

```
int readings1[numReadings];
```

```
int readings2[numReadings];
int readings3[numReadings];
int readings4[numReadings];
int readings5[numReadings];
int readings6[numReadings];
int readings7[numReadings];
int readings8[numReadings];
int readings9[numReadings];
```

//counter used for filling the smoothing function Array

شمارنده مورد استفاده در تابع برای به ترتیب پر کردن آرایه شامل داده های ورودی.

```
int readIndex1 = 0;
int readIndex2 = 0;
int readIndex3 = 0;
int readIndex4 = 0;
int readIndex5 = 0;
int readIndex6 = 0;
int readIndex7 = 0;
int readIndex8 = 0;
int readIndex9 = 0;
```

//Sum of Smoothing function Array

مجموع داده های موجود در آرایه ای که داده های ورودی در آن قرار می گیرند.

```
int total1 = 0;
int total2 = 0;
int total3 = 0;
```

```
int total4 = 0;
int total5 = 0;
int total6 = 0;
int total7 = 0;
int total8 = 0;
int total9 = 0;

//Output of Smoothing function
```
میانگین داده های ورودی.
```
int average1 = 0;
int average2 = 0;
int average3 = 0;
int average4 = 0;
int average5 = 0;
int average6 = 0;
int average7 = 0;
int average8 = 0;
int average9 = 0;

//Sensor range used for calibration
```
در این قسمت باید مینیمم و ماکسیمم عددی که سنسور در حالت طبیعی خم شدنش به عنوان ورودی به سیستم می دهد را قرار داد. برای پیدا کردن این اعداد از قسمتی از همین برنامه استفاده می شود.
```
int min1=700;
int max1=900;

int min2=700;
```

```
int max2=900;

int min3=700;
int max3=900;

int min4=700;
int max4=900;

int min5=700;
int max5=900;

int min6=700;
int max6=900;

int min7=800;
int max7=1000;

int min8=700;
int max8=900;

int min9=700;
int max9=900;


void setup()
{
```

```
//Start serial comunication for analysing sensor data
```

این خط یک ارتباط سریال را راه اندازی می کند که به ما برای خواندن داده های خام سنسور و گرفتن خروجی از قسمت های مختلف کمک می کند.

```
Serial.begin(9600);
```

```
//Pin definition for arm servo
```

در این قسمت پین هایی را که سروو موتور ها سیگنال موقعیت خود را از آنها دریافت می کنند مشخص می کنیم.

```
salman1.attach(39);
salman2.attach(37);
salman3.attach(35);
salman4.attach(34);
```

```
//Pin definition for hand servo
salman5.attach(22);
salman6.attach(24);
salman7.attach(26);
salman8.attach(28);
salman9.attach(30);
```

```
//Set the Smoothing function Array to zero
```

این قسمت برای صفر کردن آرایه ذخیره سازی داده در فیلتر مورد بحث استفاده می شود. در واقع در هنگام راه اندازی این قسمت داده های قبلی را پاک می کند.

```
for (int thisReading = 0; thisReading < numReadings; thisReading++)
{
```

```
readings1[thisReading] = 0;
readings2[thisReading] = 0;
readings3[thisReading] = 0;
readings4[thisReading] = 0;
readings5[thisReading] = 0;
readings6[thisReading] = 0;
readings7[thisReading] = 0;
readings8[thisReading] = 0;
readings9[thisReading] = 0;
}
}

void loop()
{
// // for testing with serial
```

این قسمت از برنامه در حالت نهایی غیر فعال خواهد بود و در مرحله تست موتور ها برای موقعیت دادن به هر یک از موتور ها به صورت دستی مورد استفاده قرار می گیرد. به طوری که ۹ عدد زاویه در قسمت مربوط به ارتباط سریال توسط کاربر تایپ شده و موتور های مربوط به هر مفصل به ترتیب در این ۹ موقعیت قرار می گیرند.

نهایتا از قسمت دوم همین کد می توان برای خواندن اطلاعات خام سنسور ها برای به دست آوردن حدود ماکسیمم و مینیمم استفاده کرد.

```
//while (Serial.available()>0)
//{
//   a = Serial.parseInt();
//   b = Serial.parseInt();
//   c = Serial.parseInt();
```

```
//   d = Serial.parseInt();
//
//   e = Serial.parseInt();
//   f = Serial.parseInt();
//   g = Serial.parseInt();
//   h = Serial.parseInt();
//   i = Serial.parseInt();
//  // e = Serial.parseInt();
//  if (Serial.read() == '\n')
//  {
//    a=constrain(a,10,170);
//    b=constrain(b,10,170);
//    c=constrain(c,10,170);
//    d=constrain(d,10,170);
//
//    e=constrain(e,10,170);
//    f=constrain(f,10,170);
//    g=constrain(g,10,170);
//    h=constrain(h,10,170);
//    i=constrain(i,14,170);
//
//
//   // e=constrain(e,10,170);
//  Serial.print("a:"); Serial.println(a);
//  Serial.print("b:"); Serial.println(b);
//  Serial.print("c:"); Serial.println(c);
//  Serial.print("d:"); Serial.println(d);
```

```
//
//  Serial.print("e:"); Serial.println(e);
//  Serial.print("f:"); Serial.println(f);
//  Serial.print("g:"); Serial.println(g);
//  Serial.print("h:"); Serial.println(h);
//  Serial.print("i:"); Serial.println(i);
//  Serial.print("////////////////////////////");
//
//
//  salman1.write(a);
//  salman2.write(b);
//  salman3.write(c);
//  salman4.write(d);
//
//  salman5.write(e);
//  salman6.write(f);
//  salman7.write(g);
//  salman8.write(h);
//  salman9.write(i);
//
//
//
//  }
//}
```

```
  //Reading sensor value
```

این قسمت داده های سنسور ها را می خواند.

```
 a=analogRead(A15);
 b=analogRead(A14);
 c=analogRead(A13);
 d=analogRead(A12);
 e=analogRead(A11);
 f=analogRead(A10);
 g=analogRead(A9);
 h=analogRead(A8);
 i=analogRead(A7);

//Serial.print("a:"); Serial.println(a);
//  Serial.print("b:"); Serial.println(b);
  //Serial.print("c:"); Serial.println(c);
//  Serial.print("d:"); Serial.println(d);
//
//  Serial.print("e:"); Serial.println(e);
//  Serial.print("f:"); Serial.println(f);
//  Serial.print("g:"); Serial.println(g);
//  Serial.print("h:"); Serial.println(h);
//  Serial.print("i:"); Serial.println(i);
//  Serial.println("////////////////////////////");



 //Data smoothing
```

از این قسمت به عنوان ورودی توابع فیلتر استفاده می شود.

```
a=smooth1(a);
b=smooth2(b);
c=smooth3(c);
d=smooth4(d);
e=smooth5(e);
f=smooth6(f);
g=smooth7(g);
h=smooth8(h);
i=smooth9(i);
```

```
//map sensor data to servo position
```

تابع دیگری در برنامه نوشته شده وجود دارد که وظیفه دارد عدد خارج شده از فیلتر را به عدد متناسب با زاویه موتور تغییر دهد. این قسمت به عنوان ورودی این تابع مورد استفاده قرار می گیرد.

```
a=mapping(a,min1,max1,40,120);
b=mapping(b,min2,max2,40,120);
c=mapping(c,min3,max3,40,120);
d=mapping(d,min4,max4,40,120);
e=mapping(e,min5,max5,40,120);
f=mapping(f,min6,max6,40,120);
g=mapping(g,min7,max7,40,120);
h=mapping(h,min8,max8,40,120);
i=mapping(i,min9,max9,40,120);
```

```
//Create servo signal
```

نهایتا بعد از به دست آوردن زاویه مناسب برای هر موتور این قسمت عدد زاویه را به سیگنال موقعیت مناسب برای هر موتور تبدیل می کند.

```
 salman1.write(a);
 salman2.write(b);
 salman3.write(c);
 salman4.write(d);

 salman5.write(e);
 salman6.write(f);
 salman7.write(g);
 salman8.write(h);
 salman9.write(i);


}
```

//maping function

تابعی که نقش تبدیل داده های فیلتر شده به عدد زاویه را بر عهده دارد.

```
int mapping(int x1,int x2, int x3,int x4, int x5)
{
x1=map(x1,x2,x3,x4,x5);
x1=constrain(x1,x4,x5);
return x1;
}
```

//Smoothing functions

توابعی که نقش فیلتر کردن داده ها را بر عهده دارند.

```
int smooth1(int x)                    //smoothing sensor 1 data
{
 total1= total1 - readings1[readIndex1];
  readings1[readIndex1] = x;
  total1= total1 + readings1[readIndex1];
  readIndex1 = readIndex1 + 1;
  if (readIndex1 >= numReadings)
  readIndex1 = 0;
  average1 = total1 / numReadings;
  return average1;
}

int smooth2(int x)                    //smoothing sensor 2 data
{
 total2= total2 - readings2[readIndex2];
  readings2[readIndex2] = x;
  total2= total2 + readings2[readIndex2];
  readIndex2 = readIndex2 + 1;
  if (readIndex2 >= numReadings)
  readIndex2 = 0;
  average2 = total2 / numReadings;
  return average2;
}
```

```
int smooth3(int x)                    //smoothing sensor 3 data
{
 total3= total3 - readings3[readIndex3];
  readings3[readIndex3] = x;
  total3= total3 + readings3[readIndex3];
  readIndex3 = readIndex3 + 1;
  if (readIndex3 >= numReadings)
  readIndex3 = 0;
  average3 = total3 / numReadings;
  return average3;
}

int smooth4(int x)                    //smoothing sensor 4 data
{
 total4= total4 - readings4[readIndex4];
  readings4[readIndex4] = x;
  total4= total4 + readings4[readIndex4];
  readIndex4 = readIndex4 + 1;
  if (readIndex4 >= numReadings)
  readIndex4 = 0;
  average4 = total4 / numReadings;
  return average4;
}

int smooth5(int x)                    //smoothing sensor 5 data
{
 total5= total5 - readings5[readIndex5];
```

```
 readings5[readIndex5] = x;
 total5= total5 + readings5[readIndex5];
 readIndex5 = readIndex5 + 1;
 if (readIndex5 >= numReadings)
 readIndex5 = 0;
 average5 = total5 / numReadings;
 return average5;
}

int smooth6(int x)                    //smoothing sensor 6 data
{
 total6= total6 - readings6[readIndex6];
 readings6[readIndex6] = x;
 total6= total6 + readings6[readIndex6];
 readIndex6 = readIndex6 + 1;
 if (readIndex6 >= numReadings)
 readIndex6 = 0;
 average6 = total6 / numReadings;
 return average6;
}

int smooth7(int x)                    //smoothing sensor 7 data
{
 total7= total7 - readings7[readIndex7];
 readings7[readIndex7] = x;
 total7= total7 + readings7[readIndex7];
 readIndex7 = readIndex7 + 1;
```

```
  if (readIndex7 >= numReadings)
  readIndex7= 0;
  average7 = total7 / numReadings;
  return average7;
}

int smooth8(int x)                    //smoothing sensor 8 data
{
 total8= total8 - readings8[readIndex8];
  readings8[readIndex8] = x;
  total8= total8 + readings8[readIndex8];
  readIndex8 = readIndex8 + 1;
  if (readIndex8 >= numReadings)
  readIndex8 = 0;
  average8 = total8 / numReadings;
  return average8;
}

int smooth9(int x)                    //smoothing sensor 9 data
{
 total9= total9 - readings9[readIndex9];
  readings9[readIndex9] = x;
  total9= total9 + readings9[readIndex9];
  readIndex9 = readIndex9 + 1;
  if (readIndex9 >= numReadings)
  readIndex9 = 0;
  average9 = total9 / numReadings;
```

```
  return average9;
}
```

۴. جایگذاری سنسور ها در لباس ثبت کننده حرکات دست انسان

بعد از انجام تنظیمات مربوط به نرم افزار و انجام تست های اولیه بر روی سنسور ها، می توان سنسور ها را در قسمت های مختلف دست متناسب با حرکات مورد نیاز قرار داد. بعد از این کار لازم است مرحله کالیبراسیون مجددا انجام شود تا حرکات انجام شده توسط بازو دقیقا مشابه حرکات انجام شده توسط دست در بر گرفته شده توسط لباس باشد.

۵. پیشنهاد کاربرد های جدید و مدل های آینده

این قسمت به طور تدریجی به سایت اضافه خواهد شد.