DETEKSI WARNA LAPANGAN KRSI MENGGUNAKAN KAMERA DAN SENSOR WARNA UNTUK MENCARI SUDUT HADAP DAN JARAK

SKRIPSI

TEKNIK ELEKTRO KONSENTRASI ELEKTRONIKA

Ditujukan untuk memenuhi persyaratan

memperoleh gelar Sarjana Teknik

DISUSUN OLEH :

AKHMAD TEGAR FAREZA FITDRA

NIM. 115060300111003

KEMENTERIAN RISET, TEKNOLOGI DAN PENDIDIKAN TINGGI

FAKULTAS TEKNIK

UNIVERSITAS BRAWIJAYA

MALANG

2015

LEMBAR PENGESAHAN

DETEKSI WARNA LAPANGAN KRSI MENGGUNAKAN KAMERA DAN SENSOR WARNA UNTUK MENCARI SUDUT HADAP DAN JARAK

SKRIPSI

TEKNIK ELEKTRO KONSENTRASI ELEKTRONIKA

Ditujukan untuk memenuhi persyaratan

memperoleh gelar Sarjana Teknik

Disusun Oleh :

AKHMAD TEGAR FAREZA FITDRA

NIM. 115060300111003-63

Skripsi ini telah direvisi dan disetujui oleh dosen pembimbing

pada tanggal 28 Juli 2015

Dosen Pembimbing I	Dosen Pembimbing II
Nanang Sulistiyanto, Ir., MT.	**Eka Maulana, ST., MT., M.Eng**
NIP. 19700113 199403 1 002	**NIK. 841130 06 1 1 0280**

PENGANTAR

Alhamdulillah, puji dan syukur penulis panjatkan kepada Allah SWT, karena atas segala petunjuk dan nikmat-Nya lah skripsi ini dapat diselesaikan.

Skripsi berjudul "Deteksi Warna Lapangan KRSI Menggunakan Kamera Dan Sensor Warna Untuk Mencari Sudut Hadap Dan Jarak" ini disusun untuk memenuhi sebagian persyaratan memperoleh gelar Sarjana Teknik di Jurusan Teknik Elektro Universitas Brawijaya.

Penulis menyadari bahwa dalam penyusunan skripsi ini tidak terlepas dari bantuan berbagai pihak. Oleh karena itu, dengan ketulusan dan kerendahan hati penulis menyampaikan terima kasih kepada:

- Ayah dan Ibu atas segala nasehat, kasih sayang, perhatian dan kesabarannya didalam membesarkan dan mendidik penulis, serta telah banyak mendoakan kelancaran penulis hingga terselesaikannya skripsi ini.
- Bapak M. Aziz Muslim, ST., MT., Ph.D. selaku Ketua Jurusan Teknik Elektro Universitas Brawijaya dan Bapak Hadi Suyono, ST., MT., Ph.D. selaku Sekretaris Jurusan Teknik Elektro Universitas Brawijaya.
- Bapak Rudy Yuwono, S.T., M.Sc. selaku dosen pembimbing akademik atas segala bimbingan, nasehat dan motivasi yang telah diberikan.
- Bu Nurrusa'adah selaku Dosen Pembimbing Paket B dan kedua dosen Pembimbing skripsi ini Pak Nanang Sulistiyanto dan Pa Ekan Maulana atas segala bimbingan, kritik, dan saran yang telah diberikan.
- Seluruh tim dosen dan karyawan Teknik Elektro Universitas Brawijaya atas segala bantuan, bimbingan pengalaman, dan ilmu pengetahuan yang telah diberikan.
- Seluruh anggota tim robot KRSBI atas segala dukungan dan bantuan yang telah diberikan selama ini, yaitu pada Hasdi selaku kapten tim yang baru, Chandra, Hemi, Andi, dan tidak lupa Gabi sebagai mantan kapten yang juga telah membantu banyak bagi kami.
- Seluruh anggota tim robot KRSI terutama Emon dan Bustanul yang telah menemani selama 3 tahun di tim robot, juga pada Mas Abu, Mas Tansu, Mas Muammmar, Roni, Firman, dan Gusgus yang telah banyak membantu selama 2 tahun. Sesungguhnya prestasi yang telah kita raih adalah hasil perjuangan

kita bersama. Dan juga adek-adek 2013, Surya, Yuda, dan Itsna saya ucapkan banyak terima kasih.

- Seluruh anggota tim robot TEUB atas segala motivasi dan bantuan yang diberikan, terutama perihal peminjaman alat.
- Seluruh teman-teman TEUB INVERTER yang telah menemani dan saling tolong menolong selama 4 tahun lamanya.
- Mas-mas, mbak-mbak, dan teman-teman lab sisdig angkatan 2008 hingga 2013, terima kasih atas ilmu yang bernilai besar yang telah dibagikan.
- Semua rekan-rekan sesama pekerja skripsi paket B 2011, terutama Naufal, Rizal, Swaraka, Agung, dan Nurdin sesama rekan seperjuangan dengan dosen pembimbing yang sama
- Dan semua orang yang telah membantu dan tidak bisa penulis sebutkan satu persatu, terimakasih banyak atas semua bantuannya.

Pada akhirnya, penulis menyadari bahwa skripsi ini masih belum sempurna. Oleh karena itu, penulis sangat mengharapkan kritik dan saran yang membangun. Penulis berharap semoga skripsi ini dapat bermanfaat bagi pengembangan ilmu pengetahuan dan teknologi serta bagi masyarakat.

Malang, 07 Juni 2015

Penulis

Daftar Isi

Daftar Gambar

Daftar Tabel

Ringkasan

Akhmad Tegar Fareza Fitdra, Jurusan Teknik Elektro Fakultas Teknik Universitas Brawijaya, Juni 2015, *Deteksi Warna Lapangan KRSI Menggunakan Kamera Dan Sensor Warna Untuk Mencari Sudut Hadap Dan Jarak*, Dosen Pembimbing : Nanang Sulistiyanto dan Eka Maulana.

Kontes Robot Seni Indonesia (KRSI) merupakan suatu kompetisi dimana peserta diwajibkan membuat suatu robot humanoid yang dapat menari mengikuti irama musik dan berjalan melalui lapangan perlombaan dari zona awal hingga zona akhir. Agar robot mampu berjalan menyelesaikan lapangan perlombaan sesuai dengan jalur yang diinginkan dibutuhkan suatu pedoman navigasi pada robot. Oleh karena itu, dirancanglah suatu sistem deteksi warna lapangan perlombaan menggunakan kamera dan sensor warna untuk menentukan jarak dan arah hadap, serta warna pada zona robot berada dengan aktuator penggerak berupa motor DC yang digunakan untuk menghitung jarak tempuh dengan umpan balik rotary encoder.

Aplikasi penelitian ini akan diterapkan pada robot humanoid pada perlombaan KRSI. Berbeda dengan robot yang digunakan pada perlombaan KRSI, robot yang digunakan pada penelitian ini adalah robot beroda. Robot memiliki 2 buah roda belakang sebagai roda utama dan 2 buah roda depan sebagai roda bebas. Robot memiliki tinggi sebesar 50,5 cm. Tinggi robot disesuaikan dengan peraturan perlombaan yang mengharuskan robot memiliki tinggi minimum 50 cm dan maksimum 60 cm. Robot juga dilengkapi dengan 2 buah aktuator kepala yaitu motor servo seperti robot humanoid untuk gerakkan mengangguk dan menggeleng. Roda belakang berfungsi sebagai aktuator penggerak pada piringan rotary encoder yang dipasang tepat bersebelahan dengan roda sebagai sensor putaran untuk perhitungan perkiraan jarak tempuh. Robot dilengkapi dengan kamera sebagai alat untuk melakukan pengukuran jarak tempuh berdasarkan perubahan besar sudut aktuator yang dipasang pada kepala pada saat kamera mendeteksi warna zona pada lapangan dan besar tinggi robot memanfaatkan persamaan trigonometri. Robot juga dilengkapi dengan sensor warna sebagai alat untuk mendeteksi warna tepat tegak lurus di bagian bawah robot. Hal ini dikarenakan aktuator kepala pada robot tidak mampu menggerakkan kamera hingga mendeteksi warna zona di bawah robot. Mini komputer digunakan sebagai pemroses data kamera sedangkan mikrokontroler ARM digunakan sebagai pemroses data sensor dan pengontrol aktuator. Kamera diletakkan berpasangan dengan aktuator kepala robot untuk melakukan proses deteksi warna sedangkan sensor warna diletakkan di bagian bawah robot dengan jarak 5 cm di atas permukaan tanah. Mini komputer menggunakan aturan warna HSV untuk membedakan warna yang dideteksi kamera dengan warna lain pada lapangan perlombaan. Hal ini dikarenakan warna HSV lebih mudah digunakan dalam menentukan nilai warna bagi pengguna dibandingkan warna RGB.

Berdasarkan pengujian yang dilakukan didapatkan pembacaan jarak memanfaatkan kamera dan perubahan besar sudut aktuator kepala memiliki persen error rata-rata tertinggi sebesar 15,05% pada jarak 16,45 cm dan perubahan besar sudut sendi kepala pada sudut 70°. Berdasarkan hasil pengujian pembacaan warna oleh sensor warna didapatkan keberhasilan pembacaan warna oleh sensor warna sebesar 100%. Kemudian perhitungan jarak tempuh robot memiliki persen error

rata-rata tertinggi sebesar 3,79% dengan kesalahan jarak tempuh rata-rata sebesar 18,33 mm. Hal ini disebabkan kurangnya kemampuan pengereman pada motor dan resolusi sensor putaran yang kurang baik. Pengujian pembacaan arah hadap didapatkan kesalahan rata-rata sebesar 1,52°. Semakin dekat jarak robot dengan warna yang dideteksi oleh kamera maka kesalahan pembacaan sensor akan semakin besar, hal ini disebabkan kamera yang digunakan sangat terpengaruh oleh pencahayaan ruang sehingga diperlukan kalibrasi kamera sebelum digunakan.

Kata kunci **:** KRSI, kamera, sensor warna, *trigonometri.*

SUMMARY

Akhmad Tegar Fareza Fitdra, Department of Electrical Engineering, Faculty of Engineering, University of Brawijaya, June 2015, *KRSI Arena Color Detection Using Camera And Color Sensor To Determine The Facing Angle And The Distance*, Academic Supervisor : Nanang Sulistiyanto and Eka Maulana.

Indonesian's Robotic Art Competition (KRSI) is a competition where the competitor must make a dancer humanoid robot to dance following the music's rythem and they have to walk from the arena's start zone to the end zone. So the robot can walk to finish the arena excactly as the way that the creator wanted, there is a need for a navigation method. Because of that need, it is designed a color detection system using camera and color sensor to determine the facing angle and distance of the robot to the detected color with a servo actuator as the robot's head and movement distance by using rotating sensor rotary encoder with DC motor as the wheels's actuator.

The application of this research will be used to the humanoid robot in KRSI contest, but unlike the robot in the contest, the robot in this research is a wheeled robot. The robot has 2 wheels behind and 2 upfront. The robot has a height of 50.5 cm to follow the actual rule of the contest that making the robot has a height of minimum and maksimum are 50 cm and 60 cm. The robot is equipped with 2 servo actuator as the head. The back wheels is functioned as the movement actuator for the rotary encoder that installed beside the wheels to measure the estimated movement distance because the robot will not actually move on the arena and that is not the focus of this research. A camera is installed on the robot as sensor to detect zone's color and to measure the distance from the robot present position to the detected zone using trigonometri equation. The robot is also equipped with color sensor that is used to detect the color under the robot. That is because the head's actuator can't moved the camera to detect the zone exactly perpendicular under the robot to know which zone is the robot currently at. Mini computer is used as the camera's data processor and ARM microcontroller is used in the other sensor's data processing and as the actuator's controller. The camera is installed at the head connected to the servo actuator and the color sensor is located under the robot 5 cm from the arena's surface. The mini computer is using a HSV color's rule to distinguish the camera's detected color with the other color on the arena. HSV is used because it is easier to determine the color value of object using the HSV method than using the RGB method.

Based on the experiment, result showed that the highest error persentage of the measured distanced of the detected zone using camera is 15.05% at the distance of 16.45 cm and the head's joint angle change of 70°. Result showed that the reading of the color sensor to determine the color of the zone has a successfull rate of 100%. Then the highest average error percentage of measured estimated movement distance is 3,79% and the average error is 18.33 mm. It is caused by the low resolution of the sensor and the low braking ability of the DC motor used. The average error result of the facing angle is 1.52°. With the decreement of the distance from the detected zone and the robot initial position, the error of the

sensor’s measurement will increased. It is caused by the camera that is highy influenced by the room’s lighting as the angle of head’s actuator change. Because of that reason, it is recommended to calibrate the sensor before used.

Keywords : KRSI, camera, color sensor, trigonometri

BAB I

PENDAHULUAN

1.1 Latar Belakang

Kontes robot merupakan perlombaan yang telah ada sejak dulu. Kontes Robot Nasional (KRN) atau sekarang telah berganti nama menjadi Kontes Robot Indonesia (KRI) merupakan kontes yang diselenggarakan oleh Direktorat Jendral Pendidikan Tinggi (Dirjen Dikti). KRI dibagi menjadi 5 kategori, yakni Kontes Robot ABU Indonesia (KRAI), Kontes Robot Pemadam Api Indonesia (KRPAI) Beroda dan Berkaki, Kontes Robot Sepak Bola Indonesia (KRSBI), serta Kontes Robot Seni Indonesia (KRSI). Perlombaan ini dibagi menjadi 2 tingkat, yakni tingkat Regional dan Nasional. Hanya tim yang berprestasi pada tingkat Regional yang berhak tampil di kompetisi tingkat Nasional.

Kontes Robot Seni Indonesia merupakan kontes robot yang mengharuskan peserta membuat robot humanoid yang nantinya akan menari mengikuti irama musik. Tema perlombaan KRSI selalu berubah setiap tahunnya sehingga jenis tarian yang akan dilombakan juga berubah. Penentuan pemenang perlombaan KRSI ditentukan oleh penilaian juri. Beberapa aspek yang dijadikan dasar penilaian juri, antara lain keindahan gerakan, kemampuan mendeteksi suara dan mengikuti irama musik, serta kemampuan robot humanoid untuk berjalan melalui lapangan perlombaan dan zona awal hingga akhir. Seluruh besar penilaian merupakan wewenang juri dan tidak dipengaruhi oleh peserta.

Robot humanoid memiliki kecenderungan berjalan yang tidak stabil dikarenakan berbagai faktor, diantaranya pembuatan mekanik yang tidak presisi, kondisi lapangan perlombaan yang tidak rata, serta tidak adanya feedback sensor yang mendukung pergerakan robot. Pada saat perlombaan, biasanya tim KRSI Teknik Elektro Universitas Brawijaya meletakkan robot dengan sudut hadap menjauhi titik tegak lurus dari zona mulai dengan harapan robot bisa mencapai zona akhir tanpa melakukan *retry*. *Retry* merupakan aturan yang mewajibkan peserta mengambil robot dan meletakkannya kembali di lapangan sesuai petunjuk juri. Apabila robot keluar lapangan, maka peserta wajib melakukan *retry* yang berakibat pengurangan poin yang cukup besar. Hal ini menyebabkan *feedback* sensor sangat diperlukan sebagai pedoman robot agar tidak menjauhi zona akhir.

Maka, pada skripsi ini akan dirancang suatu sistem pendeteksi warna pada robot menggunakan sensor kamera dan sensor warna sebagai metode bagi robot penari agar bisa menari dengan lancar dari zona mulai menuju zona akhir tanpa melakukan retry dan memperoleh poin maksimal untuk memenangkan perlombaan. Penggunaan sensor kamera sebenarnya sebelumnya juga pernah diteliti oleh Gladi (Buana, 2013) yang menggunakan kamera CMUCAM4 untuk mengetahui warna zona pada lapangan. Namun peneliti menyadari bahwa masih ada kekurangan pada kamera yang digunakan karena memiliki resolusi cukup rendah yaitu 160 x 120 pixel dibandingkan kamera Logitech yang digunakan pada skripsi ini yaitu 1080 x 720 pixel sehingga memiliki tingkat akurasi pembacaan warna yang lemah. Selain itu penggunaan kamera masih bisa dikembangkan untuk mengetahui jarak tempuh yang dibutuhkan robot dan diharapkan mampu membantu mengarahkan robot ke zona yang dituju. Dengan adanya sensor warna yang ada di robot diharapkan robot bisa mengetahui posisi robot berdiri.

1.2 Rumusan Masalah

Berdasarkan latar belakang dapat disusun rumusan masalah sebagai berikut:

a) Bagaimana merancang dan membuat robot yang dapat mendeteksi warna dan mengukur jarak dari posisi awal ke warna terdeteksi menggunakan kamera.
b) Bagaimana mengetahui posisi dimana robot berdiri menggunakan sensor warna.
c) Bagaimana jarak zona terdeteksi dan sudut hadap robot memanfaatkan kamera dan perubahan besar sudut aktuator kepala.
d) Bagaimana mengetahui seberapa jauh jarak tempuh robot setelah mengetahui jarak terhitung dengan umpan balik sensor putaran.

1.3 Batasan Masalah

Dengan mengacu pada permasalahan yang telah dirumuskan, maka hal-hal yang berkaitan dengan alat akan diberi batasan sebagai berikut :

a) Robot menggunakan roda.
b) Jumlah robot hanya satu.
c) Robot tidak menggunakan sistem pendeteksi musik.
d) Robot tidak melakukan pergerakan, dan pengujian jarak hanya berdasarkan perputaran roda dan umpan balik sensor putaran.

e) Mikrokontroler yang digunakan ada 2, yaitu sebagai kontroler kamera dan kontroler utama.
f) Parameter yang diperhitungkan hanya perubahan sudut servo kepala dan warna yang dideteksi oleh kamera.
g) Pembahasan mekanik robot hanya sekedar aspek praktis, tanpa membahas mekanika secara mendalam.
h) Sensor kamera dan warna yang digunakan harus dilakukan kalibrasi terlebih dahulu karena mudah terpengaruh oleh pencahayaan ruang.

1.4 Tujuan

Tujuan dari pembuatan alat ini adalah untuk merancang dan membuat robot beroda pendeteksi warna sebagai prototype panduan perhitungan jarak dan arah hadap pada robot *humanoid* KRSI.

1.5 Sistematika Penulisan

Sistematika penulisan dalam penelitian ini sebagai berikut :

BAB I Pendahuluan

Memuat latar belakang,rumusan masalah, ruang lingkup, tujuan, dan sistematika pembahasan.

BAB II Tinjauan Pustaka

Membahas teori-teori yang mendukung dalam perencanaan dan pembuatan alat.

BAB III Metode Penelitian

Berisi tentang metode-metode yang dipakai dalam melakukan perancangan, pengujian, dan analisis data.

BAB IV Perancangan

Perancangan dan perealisasian alat yang meliputi spesifikasi, perencanaan diagram blok, prinsip kerja dan realisasi alat.

BAB V Pengujian dan Analisis

Memuat aspek pengujian meliputi penjelasan tentang cara pengujian dan hasil pengujian. Aspek analisis meliputi penilaian atau komentar terhadap hasil-hasil pengujian. Pengujian dan analisis ini terhadap alat yang telah direalisasikan berdasarkan masing-masing blok dan sistem secara keseluruhan.

BAB VI Kesimpulan dan Saran

Memuat intisari hasil pengujian dan menjawab rumusan masalah serta memberikan rekomendasi untuk perbaikan kualitas penelitian dimasa yang akan datang.

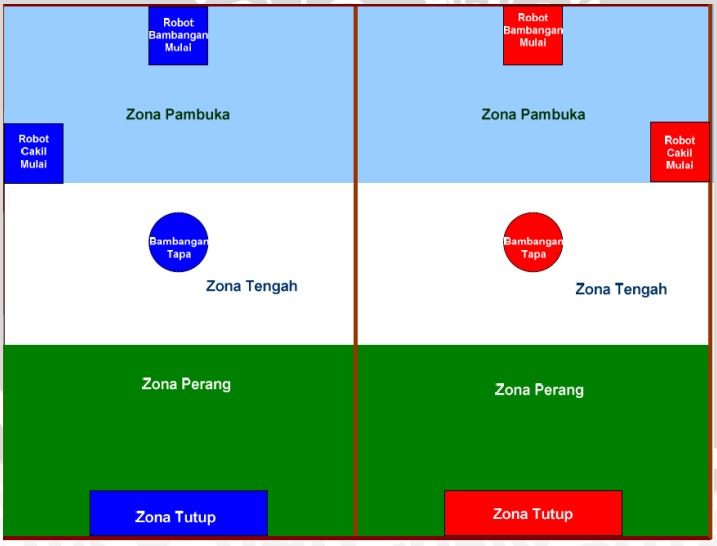
BAB 2

Tinjauan Pustaka

2.1 Kontes Robot Seni Indonesia

Kontes Robot Seni tari Indonesia (KRSI) merupakan suatu ajang kompetisi perancangan dan pembuatan robot yang disertai dengan unsur-unsur seni dan budaya bangsa yang telah terkenal di bumi pertiwi. KRSI pertamakali diadakan pada tahun 2009 yang mengangkat tema ”Robot Penari Jaipong”, tahun 2010 dengan mengangkat tema ”Robot Penari Pendet” , tahun 2011 dengan mengangkat tema ”Robot Penari Kelono Topeng”, tahun 2012 mengangkat tema “Robot Penari Piring”, tahun 2013 mengangkat tema “Robot Anoman Duto” dan pada tahun 2014 mengangkat tema “Bambangan Cakil”. Setiap tim peserta yang terdiri dari 3(tiga) mahasiswa dengan seorang dosen pembimbing, diwajibkan untuk membuat dua robot untuk menampilkan gerak seni tari budaya yang diinginkan sesuai tema kontes. (Dikti, 2014)

KRSI memiliki beberapa ketentuan umum yang berubah-ubah tiap tahunnya. Pada tahun 2015 ketentuan umum pada perlombaan KRSI, salah satunya adalah robot harus dapat menari di atas arena persegi-panjang dengan start berwarna merah atau biru. Arena berukuran (3000 x 2000) mm yang akan dibagi tiga bagian pokok.



Gambar 2.1 Lapangan perlombaan KRSI.

Sumber : Dikti (2014:23).

Pada setiap zona lapangan perlombaan, jenis gerakan tarian yang akan dilakukan robot penari berbeda-beda. Robot Humanoid diwajibkan berjalan dari zona Bambangan

Mulai menuju zona Pembuka, zona Tengah, zona Bambangan Tapa, zona Perang, dan yang terakhir zona Tutup. Khusus pada zona Bambangan Tapa, apabila robot penari gagal menginjak zona maka wajib melakukan *retry* yang berakibat pengurangan poin. *Retry* juga wajib dilakukan apabila robot terjatuh atau menabrak dinding pembatas pada pinggir lapangan.

2.2 OpenCV

OpenCV *(Open Source Computer Vision Library)* merupakan *library* pemrograman pada komputer yang bersifat *open source* dan didalamnya terdapat ratusan algoritma untuk melakukan pemrograman *computer vision. Library* ini memiliki antarmuka C, C++, phyton, dan java, serta dapat digunakan oleh windows, linux, maupun android. OpenCV didesain untuk meningkatkan efisiensi pada komputasi dan memiliki fokus pada aplikasi *real-time*. (Itseez, 2014)

Gambar 2.2 Aplikasi OpenCV dalam mendeteksi wajah pada gambar.
Sumber : Itseez (2014:373).

Beberapa fitur yang dimiliki OpenCV antara lain seperti :

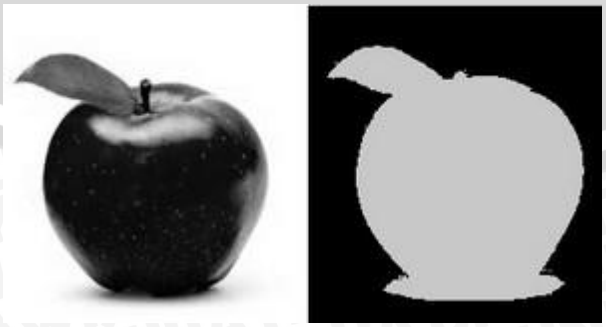
- Manipulation data citra (alokasi, *copying*, *setting*, konversi).
- Citra dan video I/O (file dan *camera based input, image/video file output*).
- Manipulasi Matriks dan Vektor beserta rutin-rutin aljabar linear (*products, solvers, eigenvalues, SVD*).
- Data struktur dinamis (*lists, queues, sets, trees, graphs*).
- Pemroses Citra fundamental (*filtering, edge detection, corner detection, sampling and interpolation, color conversion, morphological operations, histograms, image pyramids*).

- Analisis struktur(*connected components, contour processing, distance transform, various moments, template matching, Hough transform, polygonal approximation, line fitting, ellipse fitting, Delaunay triangulation*).
- Kalibrasi kamera (*calibration patterns, estimasi fundamental matrix, estimasi homography, stereo correspondence*).
- Analisis gerakan (*optical flow, segmentation, tracking*).
- Pengenalan obyek (*eigen-methods, HMM*).
- Graphical User Interface (d*isplay image/video,* penanganan keyboard dan *mouse handling, scroll-bars*).
- Pelabelan citra (*line, conic, polygon, text drawing*).

2.3 *Color Tracking*

Kamera merupakan alat yang biasa digunakan dalam pengambilan gambar. *image processing*. Image processing merupakan sebuah bentuk pemrosesan sinyal dimana inputnya adalah gambar seperti foto maupun frame video dan outputnya berupa data digital. Ada berberapa macam *image processing* salah satunya adalah *color tracking*. (Yilmaz, Javed, & Shah, 2006)

Color tracking merupakan kemampuan untuk mengambil gambar, memisahkan warna tertentu, dan mengambil informasi berupa posisi dari warna yang dipasahkan tersebut. Sebagai contoh, jika seseorang melihat sebuah foto yang berisi gambar bola merah di jalan dan ingin melingkari warna merah yang ada di dalam foto, orang tersebut akan dengan mudah melingkari bola merah di dalam gambar. Inilah ide dasar dari *color tracking*. Seseorang tidak perlu mengetahui objek yang ada pada foto adalah bola, dia hanya perlu mengetahui warna dari bola tersebut. (Adam Goode, 2011)



Gambar 2.3 Sistem color tracking dengan memisahkan 1 warna dengan warna lainnya
Sumber : Itseez (2014:210)

Salah satu cara dalam melakukan tracking warna, yaitu dengan menentukan nilai minimum dan maksimum untuk tiga jenis warna dasar. Setiap warna yang unik diwakili oleh nilai merah, hijau, dan biru yang dapat dicampur untuk menjadi warna baru. Bagian tersulit dalam menentukan warna adalah bahwa seseorang harus menentukan rentang nilai yang diijinkan untuk semua tiga jalur warna. Karena cahaya tidak sempurna seragam dan warna obyek tidak sempurna seragam, programer perlu untuk mengakomodasi untuk variasi ini. Namun, seseorang tidak akan memberi batas nilai terlalu lebar, atau banyak warna yang tidak diinginkan akan diterima.

2.4 Sensor Warna

Sensor warna yang digunakan pada alat ini adalah IC TCS3200 yang merupakan IC pengkonversi warna cahaya ke frekuensi. Silicon fotodiode dan pengkonversi arus ke frekuensi merupakan komponen utama pembentuk IC TCS3200. Sensor ini menghasilkan keluaran berupa pulsa digital sesuai dengan warna cahaya yang diterima, Sensor memiliki 3 pilihhan skala keluaran yaitu 100%, 20%, dan 2%. Skala keluaran sensor ini dapat diatur dengan mengatur keadaan 2 pin masukan sensor yaitu S0 dan S1. Sensor warna memiliki 4 macam fotodioda yaitu fotodioda dengan filter merah, fotodioda dengan filter biru, fotodioda dengan filter hijau, dan fotodioda tanpa filter (clear). Jenis fotodioda yang digunakan saat pengukuran dapat diatur dengan menetukan keadaan 2 pin masukan sensor yaitu pin S2 dan S3. Tabel pengontrolan masukan pada fotodioda dapat dilihat pada Tabel 2.1. (TAOS, 2009)

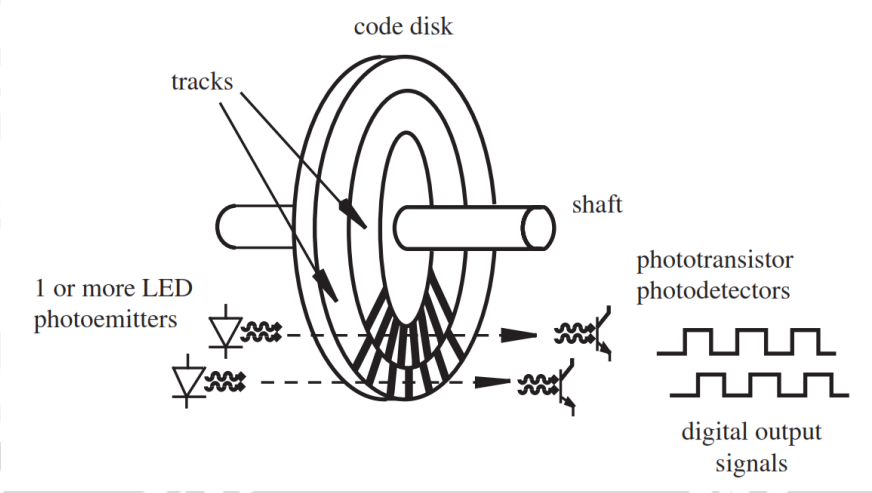
Tabel 2.1 Kombinasi S0,S1,S2,dan S3 untuk mengatur kerja sensor warna

S0	S1	Skala Frekuensi Keluaran	S2	S3	Photodiode
0	0	Power Down	0	0	Red
0	1	2 %	0	1	Blue
1	0	20 %	1	0	Clear
1	1	100 %	1	1	Green

2.5 *Rotary Encoder*

Rotary encoder digunakan untuk mendeteksi banyak putaran pada motor DC. Salah satu *rotary encoder* yang sering digunakan adalah jenis optik. Sensor ini terdiri dari *Light Emitting Diode* (LED) inframerah dan phototransistor. Secara prinsip sensor mendeteksi halangan diantara LED inframerah dan phototransistor. Apabila cahaya dari LED tidak terhalang maka cahaya dari LED akan mengenai phototransistor sehingga

phototransistor akan aktif. Apabila cahaya LED terhalang maka phototransistor tidak aktif. Diagram sensor ditunjukkan dalam Gambar 2.4.



Gambar 2.4 Rotary Encoder.

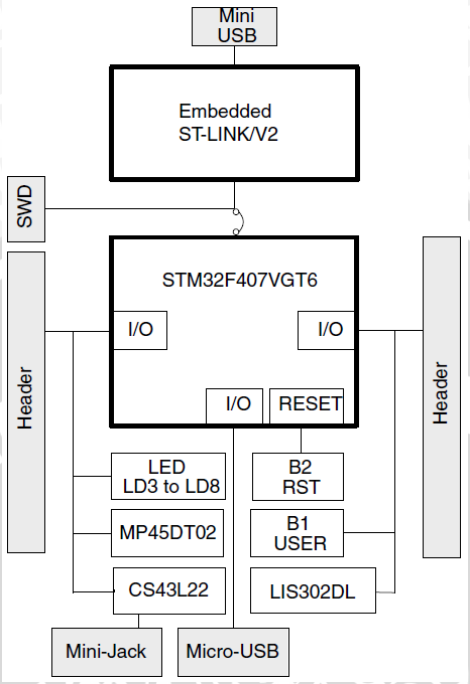
Sumber: Alciatore dan Histand (2012:384).

Incremental rotary encoder atau kadang-kadang disebut rotary encoder relatif dalam hal desain lebih sederhana dibandingkan *absolute rotary encoder*. *Incremental rotary encoder* yang memiliki keluaran sinyal quadrature terdiri dari dua LED dan dua phototransistor dan memiliki keluaran yang disebut keluaran A dan B. Saat porosnya diputar, sensor mengeluarkan pulsa yang frekuensinya sebanding dengan kecepatan rotasi poros. Pola keluaran pada saluran A dan B ditunjukkan dalam Gambar 2.3. Dengan menghitung jumlah pulsa dan mengetahui resolusi sensor putaran, jumlah putaran dapat diukur. Keluaran A dan B digunakan untuk mengetahui arah rotasi dengan melihat saluran mana yang mendahului. (Alciatore & Histand, 2012)

2.6 Modul Mikrokontroler ARM

Salah satu mikrokontroler yang mulai banyak digunakan saat ini yaitu mikrokontroler berbasis ARM. STM32F4 Discovery merupakan board mikrokontroler berbasis ARM Cortex-M4 yang dikembangkan oleh STMicroelectronics. Modul mikrokontroler ini memiliki keunggulan dibandingkan dengan mikrokontroler lain. Keunggulan mikrokontroler ini yaitu memiliki kecepatan proses data yang sangat tinggi dan RAM yang besar sehingga dapat diaplikasikan dalam skala yang lebih luas. Modul mikrokontroler ARM ini juga memiliki beberapa perangkat tambahan yang sudah terintegrasi ke dalam modul, diantaranya ST-LINK/V2 sebagai IC untuk melakukan

programming pada modul, *ST MEMS accelerometer, Audio DAC, mini USB port*, dan *micro USB port.* (STMicroelectronics, 2012)



Gambar 2.5 Blok Diagram *Hardware* STM32F4 Discovery.

Sumber : STMicroelectronics (2012:8).

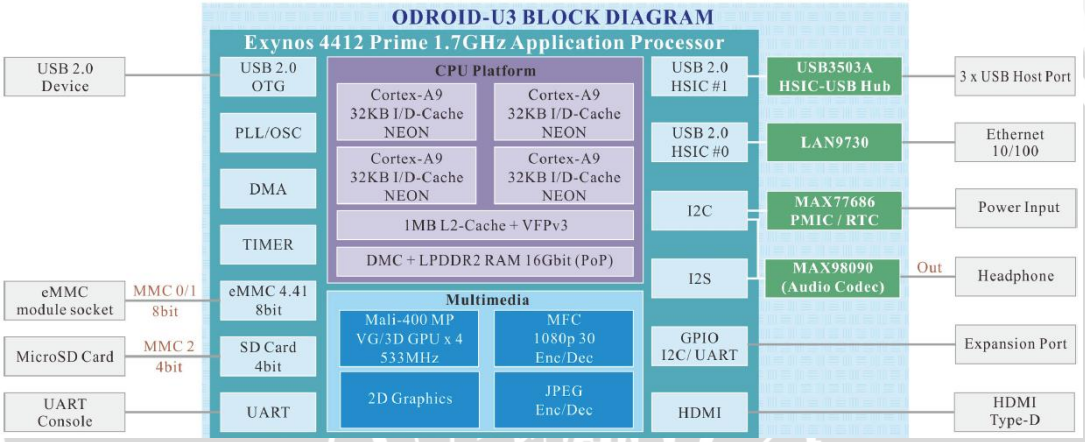
Fitur-fitur yang dimiliki STM32F4 Discovery sebagai berikut:

1. Mikrokontroler ARM Cortex M4 32 bit yang memiliki kemampuan tinggi, dengan daya rendah dengan kecepatan hingga 168 MHz.
2. Memiliki kapasitas *flash* memori 1 Mbyte, SRAM 192+4 *Kbyte*, termasuk 64 Kbyte CCM (*Core Coupled Memory*) Data RAM.
3. Saluran I/O sebanyak 140 buah dengan kemampuan *interrupt*.
4. Memiliki 15 saluran *interface* (3 saluran I2C, 4 saluran USART, 2 saluran UART, 3 saluran SPI, 2 CAN *interface*, dan SDIO *interface*).
5. Fitur *pheripheral*
   - Tujuh belas buah *Timer/Counter* dengan kemampuan pembandingan.
     - ✓ 2 (dua) buah *Timer/Counter* 16 bit
     - ✓ 16 (enam belas) buah *Timer/Counter 32* bit
   - *Real Time Counter* dengan *Oscilator* tersendiri
   - 4 *channel* PWM
   - 24 *channel*, 12-*bit* ADC
   - 12- bit DAC

- *Multi* DMA *controller*

2.7 Mini PC Odroid U3

Mini PC merupakan Personal Computer (PC) berukuran kecil yang memiliki kemampuan hampir sama dengan PC pada umumnya. Mini PC juga dapat memiliki *Operating System* di dalamnya seperti Android, Ubuntu, maupun Windows. ODROID-U3 merupakan salah satu Mini PC berkemampuan dan memiliki kecepatan tinggi. Mini komputer ini juga menawarkan banyak keuntungan dibandingkan Windows atau OSX komputer biasa, seperti *silent operation*, 5W-rata penggunaan listrik, dan portabilitas instan. (Hall, 2013)



Gambar 2.6 Blok diagram *Hardware* Odroid U3.

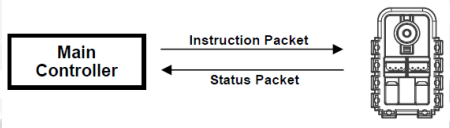
Sumber : Hall (2013:6).

Mini komputer dilengkapi dengan beberapa fitur tambahan diantaranya 3 buah port USB 2.0, 1 buah port ethernet, analog audio jack, micro usb connector, 4 buah I/O yang bisa digunakan untuk komunikasi UART atau I2C, PWM output untuk kipas pendingin, micro SD card slot, dan eMMC modul socket. Modul memiliki 2 buah led indikator sebagai penanda proses dimulainya booting. Mini komputer memiliki prosesor Exynos 4412 1.7GHz Quad-core dari Samsung dengan RAM 2GB. Ukuran komputer ini hanya 83 x 48 mm dengan berat 48g.

2.8 Servo Serial *Dynamixel*

Dynamixel adalah aktuator robot cerdas modular yang menggabungkan *gear reducer*, motor DC presisi dan sirkuit kontrol, semua dalam satu paket. Meskipun ukurannya yang ringkas, dapat menghasilkan torsi tinggi dan dibuat dengan bahan berkualitas tinggi untuk memberikan kekuatan yang diperlukan dan struktural ketahanan

untuk menahan kekuatan eksternal yang besar. Hal ini juga memiliki kemampuan untuk mendeteksi dan bertindak pada kondisi internal seperti perubahan suhu internal atau tegangan suplai. Aktuator *Dynamixel* memiliki banyak keunggulan dibandingkan produk sejenis. Posisi dan kecepatan dapat dikendalikan secara serial dengan resolusi 1024 *step*. (Robotis, 2006)



Gambar 2.7 Sistem pengontrolan dynamixel.

Sumber : Robotis (2006:9).

Kontroler utama berkomunikasi dengan unit *Dynamixel* dengan mengirim dan menerima paket data. Ada dua jenis paket; yang "*Instruction Packet*" (dikirim dari kontroler utama ke aktuator servo) dan "*Status Packet*" (dikirim dari Aktuator untuk pengendali utama). Struktur paket instruksi pada pengiriman data ke aktuator servo serial dapat dilihat pada Gambar 2.8.

OXFF	0XFF	ID	LENGTH	ERROR	PARAMETER1	PARAMETER2	...	PARAMETER N	CHECK SUM

Gambar 2.8 Paket instruksi pengiriman data pada dynamixel.

Sumber : Robotis (2006:10).

Kedua paket data pertama yaitu 0xFF menandakan awal dari paket yang akan datang. ID adalah nomor identitas yang unik dari aktuator yang berjumlah 254 ID dan memiliki range 0x00 - 0xFD. LENGTH merupakan panjang paket data dimana memiliki nilai jumlah parameter (N) + 2. INSTRUCTION merupakan instruksi yang harus dilakukan oleh aktuator. PARAMETER adalah informasi taambahan yang diberikan pada instruksi tertentu. CHECKSUM merupakan metode komputasi untuk pengecekan data. Persamaan untuk mencari nilai CHECKSUM dapat dilihat pada persamaan (2-1). Jika nilai yang terhitung lebih besar dari 255 maka bit terendah dianggap sebagai nilai CHECKSUM. Simbol "~" menandakan logika not.

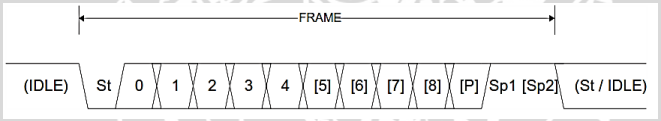
$$CHECKSUM = \sim (ID + LENGTH + INSTRUCTION + PARAMETER\ 1 + \cdots PARAMETER\ N) \quad (2\text{-}1)$$

2.9 Komunikasi Serial Asinkron

Pada prinsipnya, komunikasi serial ialah komunikasi dengan pengiriman data yang dilakukan per bit, sehingga lebih lambat dibandingkan komunikasi pararel seperti pada port printer yang mampu mengirim 8 bit sekaligus dalam sekali waktu. Devais pada

komunikasi serial port dibagi menjadi 2 (dua) kelompok yaitu *Data Communication Equipment* (DCE) dan *Data Terminal Equipment* (DTE). Pengiriman data secara serial dapat berupa sinkron, yaitu pengiriman clock dilakukan bersamaan dengan data, atau berupa asinkron, yaitu pengiriman clock tidak bersamaan dengan data namun secara dua tahap, saat data dikirim dan data diterima. Untuk istilah yang sering digunakan untuk mengirim dan menerima data secara asinkron biasa disebut *Universal Asynchronous Receiver Transmitter* (UART). Komunikasi data serial menggunakan UART sangat umum dan mudah penggunaannya, misalnya pada port serial PC. (Atmel, 2007)

Setiap pengiriman data pada UART menggunakan bit tanda start bit dan stop bit. Jalur data yang digunakan hanya satu untuk setiap pengiriman data. Data-data serial dikirim melewati jalur data satu persatu setiap satuan waktu. Paket data dimulai dengan start bit diikuti dengan *Least Significant Data Bit* (LSB). Data berikutnya adalah bit data yang dibentuk hingga 9 bit dan diakhiri *dengan Most Significant Bit* (MSB). Jika diaktifkan, maka bit parity akan diletakkan setelah bit data. Paket data diakhiri dengan stop bit. Setelah melakukan pengiriman data, maka komunikasi bisa langsung dimulai lagi atau jalur komunikasi bisa diset ke logika tinggi (idle) seperti pada Gambar 2.9.

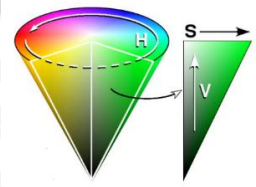


Gambar 2.9 Format Data komunikasi serial.

Sumber : Atmel (2010:148).

2.10 Warna HSV

Warna HSV (*Hue, Saturation, Value*) merupakan model pewarnaan yang mendeskripsikan warna (*hue*) dalam hal tebal warna (*saturation* atau *shade*) dan kecerahan warna (*value* atau pencahayaan). Sistem warna HSV dapat digambarkan seperti sebuah silinder atau kerucut warna. Keunggulan warna HSV dibandingkan warna RGB adalah segmentasi warna lebih mudah dilakukan pada warna HSV. Sistem kerucut warna HSV dapat dilihat dalam Gambar 2.10. (Howard, 2015)



Gambar 2.10 HSV *Color Cone*.

Sumber : Latoschik (2005:9).

BAB 3

Metode Penelitian

Penyusunan penelitian ini didasarkan dalam masalah yang bersifat aplikatif, yaitu perencanaan dan perealisasian alat agar dapat bekerja sesuai dengan yang direncanakan dengan mengacu dalam rumusan masalah. Langkah-langkah yang perlu dilakukan untuk merealisasikan alat yang dirancang adalah penentuan spesifikasi alat, pembuatan alat, dan pengujian alat

3.1 Penentuan Spesifikasi Alat

Spesifikasi alat secara global ditetapkan terlebih dahulu sebagai acuan dalam perancangan selanjutnya. Spesifikasi penting yang dicantumkan adalah yang memiliki fungsi penting pada perancangan. Spesifikasi alat yang direncanakan yaitu :

- Robot memiliki tinggi total antara 50 – 60 cm.
- Berat robot kurang dari 30 kg.
- Kamera yang digunakan adalah kamera Logitech C720HD.
- Jumlah total DOF yaitu 2 DOF.
- Motor DC yang digunakan berjumlah 2 buah dengan tipe PG-28.
- Rotary encoder sebagai sensor putaran berjumlah 2 buah menggunakan piringan roda dan sensor optokopler.
- Sumber tegangan berupa baterai *Lithium Polymer* 3S 12.6V 2200mAH.
- Motor servo yang digunakan bertipe Dynamixel AX-12.
- Sensor warna bertipe TCS3200.
- Rangkaian catu daya 5V berupa rangkaian regulator *switching* LM2576 *Simple Switcher* untuk mikrokontroler, mini komputer dan sensor, serta 12V untuk servo kepala.
- Kontroller utama menggunakan modul ARM STM32F4 Discovery.

3.2 Perancangan dan Pembuatan Alat

Pembuatan alat dalam penelitian ini terbagi menjadi dua bagian, yaitu perangkat keras dan perangkat lunak.

1) Pembuatan perangkat keras sistem menggunakan komponen-komponen mekanik dan elektronik yang sesuai sehingga dapat memenuhi spesifikasi alat yang berkaitan

dengan mekanik robot, sensor, *rotary encoder*, mikrokontroler, dan catu daya. Sistem mekanik dirancang dengan menggunakan program 3DS MAX, sedangkan sistem elektronik dirancang dengan menggunakan program Eagle.

2) Pembuatan perangkat lunak modul mikrokontroler STM32F4 Discovery menggunakan software Coocox CoIDE 1.7.0 yang memakai bahasa C.

3.3 Pengujian Alat

Untuk menganalisis kesesuaian kinerja alat dengan perencanaan, maka dilakukan pengujian sistem. Pengujian dilakukan pada masing-masing blok dan secara keseluruhan. Pengujian yang dilakukan meliputi:

1) Pengujian Sensor Kamera

Pengujian sensor kamera bertujuan untuk mengetahui apakah warna HSV yang ditangkap oleh sensor sesuai dengan warna HSV yang dibaca pada foto yang diambil menggunakan kamera yang sama. Nilai HSV pada foto dapat diketahui memanfaatkan software Paint pada PC. Nilai HSV akan digunakan sebagai parameter untuk menentukan perbedaan warna.

2) Pengujian Sensor Warna

Pengujian sensor warna dilakukan untuk mengetahui nilai keluaran sensor warna yang digunakan. Sensor memiliki keluaran berupa pulsa kotak dengan frekuensi berubah tergantung warna. Pengujian dilakukan dengan menguji 4 mode pada sensor.

3) Pengujian *Rotary Encoder*

Pengujian *rotary encoder* dilakukan untuk mengetahui apakah rotary encoder memiliki nilai keluaran dan jumlah pulsa per putaran sebagaimana mestinya. Pengujian juga meliputi pengamatan menggunakan osiloskop. Sensor akan digunakan untuk menghitung jarak tempuh berdasarkan putaran roda.

4) Pengujian Komunikasi Serial

Pengujian komunikasi serial bertujuan untuk membuktikan data serial yang dikirimkan melalui mini PC penmroses data kamera dapat diterima oleh modul mikrokontroler utama dan ditampilkan pada serial monitor dengan sempurna melalui serial monitor pada PC.

5) Pengujian Servo Serial

Pengujian ini dilakukan dengan mengamati sinyal keluaran berupa sinyal PWM penggerak motor servo untuk mengetahui apakan servo dapat bergerak dengan sudut yang sesuai dengan perhitungan berdasarkan besar data serial dan dibuktikan menggunakan

busur derajat. Data serial dikirim oleh mikrokontroler menggunakan komunikasi UART. Data serial kemudian ditampilkan pada terminal PC.

6) Pengujian Pengukuran Jarak Benda Menggunakan Trigonometri

Pengujian ini bertujuan untuk mengetahui jika robot mampu membaca jarak warna terdeteksi dari posisi robot berada memanfaatkan kamera dan perhitungan jarak menggunakan algoritma trigonometri. Parameter yang menentukan adalah besar perubahan sudut sendi kepala/ Pengukuran jarak diukur dengan menggunakan meteran.

7) Pengujian Arah Hadap Robot

Arah hadap robot diukur dengan parameter berupa besar perubahan sudut. Pengujian ini bertujuan untuk mengetahui apakah robot dapat mengetahui sudut hadap dari posisi robot berada melalui besar perubahan sudut servo pada kepala setelah melakukan pendeteksian warna. Sudut sendi kepala diukur dengan busur derajat.

8) Pengujian Pembacaan Jarak Tempuh

Pengujian ini dilakukan untuk mengetahui apakah robot mampu mengukur perubahan jarak ketika motor bergerak dengan memanfaatkan sensor *rotary encoder*. Pada saat pengujian robot diangkat dan kondisi motor dalam keadaan tanpa beban. Pengujian dilakukan pada kedua motor DC penggerak roda.

9) Pengujian Keseluruhan

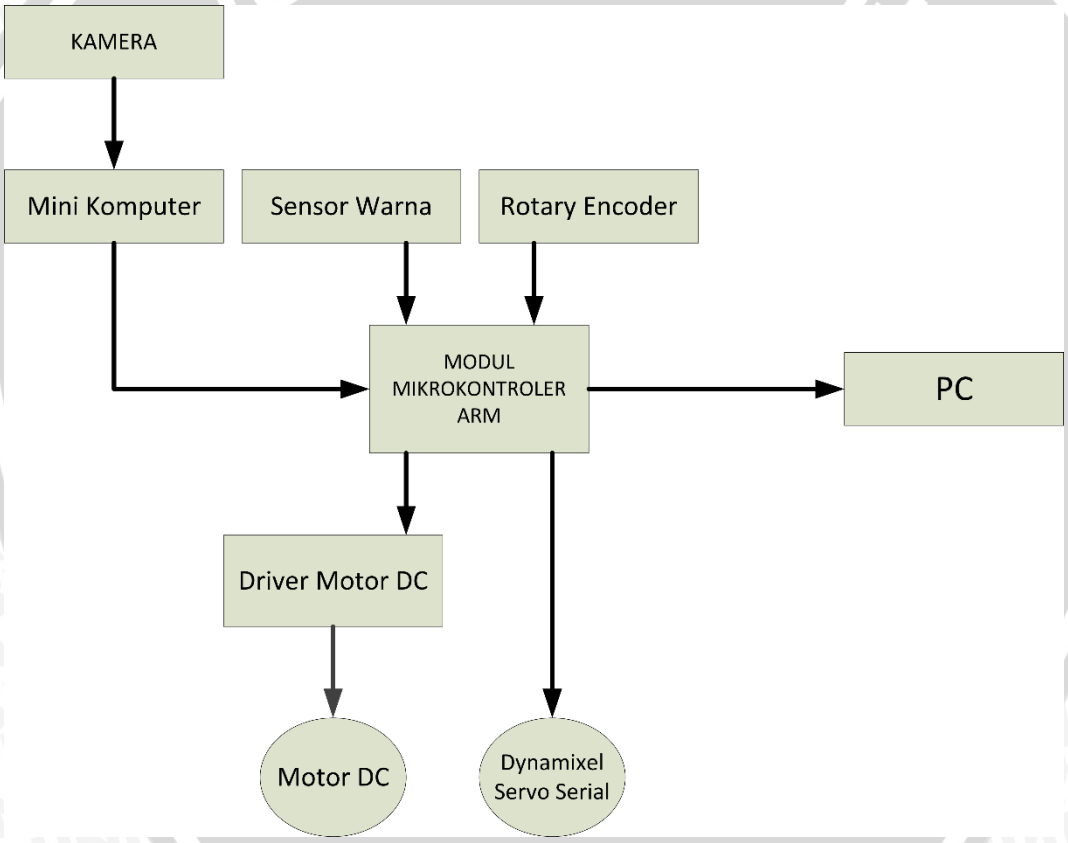
Pengujian ini dilakukan untuk mengetahui apakan seluruh komponen yang digunakan pada robot telah bekerja sebagaimana mestinya. Pengujian dilakukan pada lapangan perlombaan KRSI. Pengujian dilakukan dengan menguji keseluruhan sensor warna, kamera, sevo kepala, motor DC, dan sensor putaran.

BAB 4

Perancangan dan Pembuatan Alat

Perancangan robot ini dilakukan secara bertahap dalam bentuk blok sehingga akan memudahkan dalam analisis setiap bloknya. Perancangan terdiri dari beberapa bagian, antara lain perancangan sistem, perancangan perangkat keras, dan perancangan perangkat lunak. Perancangan perangkat keras yang terdiri dari dua bagian yaitu sistem mekanik robot dan desain sistem elektronik

4.1 Perancangan Sistem



Gambar 4.1 Blok diagram keseluruhan sistem.

Robot berfungsi untuk melakukan proses deteksi warna pada lapangan sehingga dilengkapi dengan sensor dan aktuator yang mendukung. Sebagai pengontrol sistem robot memiliki dua buah kontroler, yaitu modul STM32F4 Discovery sebagai kontroler utama dan mini PC Odroid U3 yang khusus sebagai pemroses data sensor kamera. Data dari sensor kamera akan dikirim melalui komunikasi serial UART.

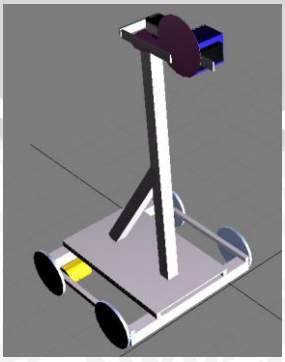
Sensor warna berfungsi sebagai pendeteksi warna zona robot berada karena kamera tidak bsa digunakan untuk mendeteksi warna tepat tegak lurus di bagian bawah robot, sedangkan rotary encoder berfungsi untuk menghitung jumlah putaran roda pada robot. Seluruh data sensor akan diproses oleh kontroler utama yang nantinya akan membuat robot bekerja sebagaimana mestinya. Proses selanjutnya adalah data yang telah diproses oleh mikrokontroler akan dikirim melalui komunikasi serial menuju PC. Karena PC yang digunakan tidak memiliki port serial maka digunakanlah modul USB *to* TTL *converter* supaya data serial dari modul mikrokontroler dapat dibaca oleh PC.

Robot memiliki 4 buah roda, yaitu 2 buah roda sebagai penggerak utama dengan aktuator berupa 2 buah motor DC berada di belakang dan 2 buah roda bebas yang dimanfaatkan sebagai penyeimbang berada di depan. Robot juga memiliki 2 *degree of freedom* (DOF) sebagai penggerak kepala robot. Aktuator yang digunakan sebagai penggerak kepala adalah motor servo serial.

4.2 Perancangan Perangkat Keras

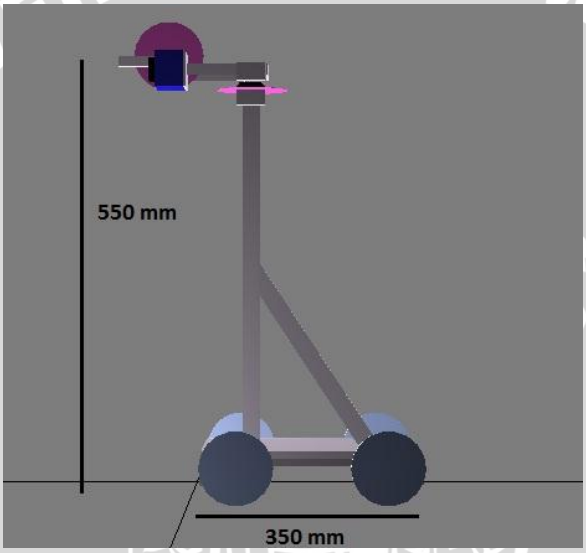
4.2.1 Perancangan Mekanik Robot

Sistem mekanik yang baik, mendukung pergerakan robot menjadi lebih baik, oleh karena itu perancangan mekanik dalam hal ini bodi dan rangka robot haruslah proporsional dengan panjang dan lebar robot. Bentuk mekanik robot beserta ukuran panjang dan lebar ditunjukkan dalam Gambar 4.2. Robot memiliki dimensi panjang 350 mm dan lebar 350 mm. Dengan besar panjang dan lebar yang sama diharapkan akan membuat robot lebih mudah dalam bernavigasi. Tinggi robot disesuaikan dengan tinggi robot yang biasa dipakai dalam perlombaan Kontes Robot Seni Indonesia yang biasanya memiliki *aturan* dimensi robot antara 500 mm – 600 mm. Keseluruhan rangka terbuat dari almunium, pemilihan almunium sebagai rangka dikarenakan sifat almunium yang relatif lebih ringan dari pada logam yang lain.

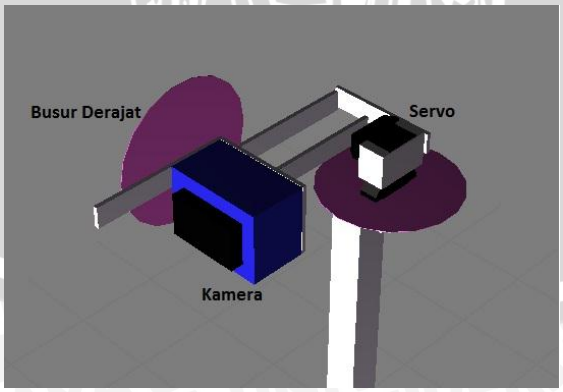


Gambar 4.2 Desain mekanik robot.

Bagian penting dari perancangan mekanik robot ini adalah penempatan sensor. Selain sensor kamera yang diletakkan pada tinggi sesuai tinggi robot pada peraturan perlombaan KRSI, sensor warna diletakkan pada jarak yang diperkirakan sesuai jika sensor diletakkan pada kaki robot *humanoid* KRSI yang sebenarnya, yaitu 50 mm - 100 mm. *Rotary encoder* dipasang tepat sejajar dengan roda penggerak yaitu ditengah badan robot. Hal ini bertujuan supaya titik robot saat pembacaan posisi berada tepat di tengah robot dan untuk menghindari slip saat robot berjalan. *Rotary encoder* disambung dengan roda berdiameter 50 mm dan berlapis spons. Untuk menjaga agar roda robot tidak slip, selain dengan cara melapisi roda dengan spons pada dudukan *rotary encoder* juga dipasang pegas. Pada setiap bagian dari DOF robot yaitu servo, dilengkapi dengan busur derajat untuk mengetahui perubahan besar sudut putaran servo seperti pada Gambar 4.4.



Gambar 4.3 Desain mekanik robot dari samping.



Gambar 4.4 Desain kepala robot.

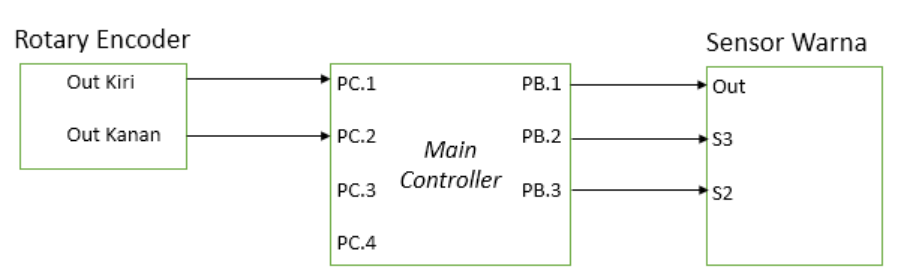
4.2.2 Perancangan Sistem Elektronik

4.2.2.1 Perancangan Sistem Antarmuka Mikrokontroler

Pada perancangan perangkat keras robot ini menggunakan modul mikrokontroller ARM sebagai pengolah utama untuk pemrosesan algoritma dan data sensor. Mikrokontroller mempunyai 6 port dan 96 saluran input/output yang dapat diprogram menjadi masukan atau keluaran. Pada perancangan ini pin-pin yang digunakan adalah:

- Pin B.1 = output sensor warna
- Pin B.2 = pin S3 pada sensor warna
- Pin B.3 = pin S2 pada sensor warna
- Pin B.6 = Pin Serial Transmitter untuk transmisi UART dengan PC
- Pin B.7 = Pin Serial Receiver untuk transmisi UART dengan PC
- Pin B.10 = Pin Serial Transmitter untuk transmisi UART dengan Mini PC
- Pin B.11 = Rx3 untuk transmisi UART dengan Mini PC
- Pin C.1 = masukan penanda arah rotary kiri (keluaran B)
- Pin C.2 = masukan penanda arah rotary kanan (keluaran B)

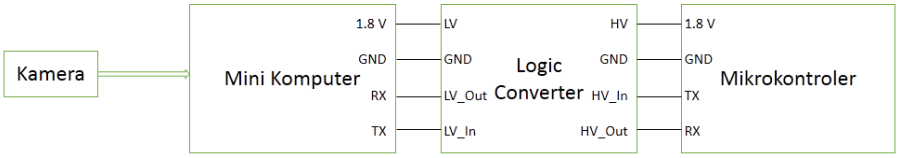
Mikrokontroler ARM merupakan kontroler utama yang nantinya akan berhubungan dengan sensor. Beberapa sensor yang digunakan antara lain sensor warna dan rotary encoder seperti pada diagram blok yang dapat dilihat pada Gambar 4.5. Mikrokontroler menerima masukkan sensor *rotary encoder* sebagai sensor putaran dan sensor warna untuk mendeteksi warna zona lapangan.



Gambar 4.5 Blok diagram antarmuka kontroler utama dengan sensor warna dan rotary encoder.

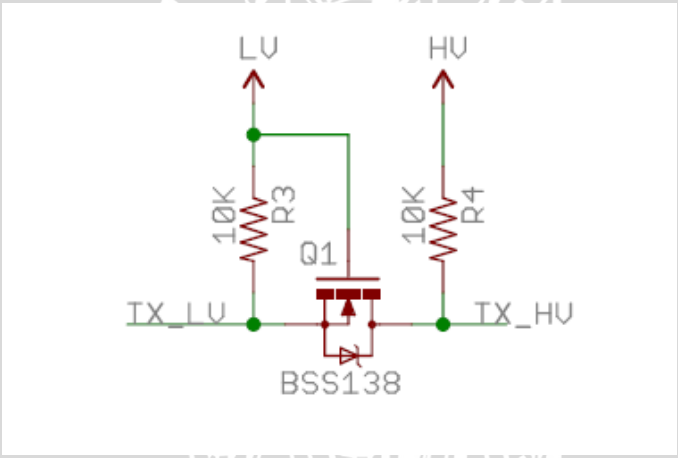
Rotary encoder yang digunakan ada 2 dimana masing masing memiliki memiliki 1 keluaran yang digunakan, yakni *out rotary encoder* kiri sebagai keluaran penghitung *rotary* kiri, *out rotary encoder* kanan sebagai keluaran penghitung *rotary* kanan. PC.1 hingga PC.2 merupakan pin masukan mikrokontroler dari *rotary encoder*. Sensor warna memiliki keluaran berupa pulsa digital dengan frekuensi berbeda-beda. Besar frekuensi berubah tergantung pada warna yang dideteksi pada lapangan. Keluaran sensor juga dapat

diatur dengan mengaktifkan fotodioda pada sensor warna secara bergantian dengan memberi masukan pada pin S_2 dan S_3 pada sensor warna.



Gambar 4.6 Blok diagram antarmuka kontroler utama dengan sensor warna dan *rotary encoder*.

Pada perancangan alat ini, robot juga dilengkapi dengan kamera. Kamera yang digunakan merupakan kamera webcam dengan koneksi berupa kabel usb. Kamera membutuhkan kemampuan pemrosesan data yang tinggi sehingga diperlukan suatu mini komputer sebagai perantara pemrosesan data kamera sebelum dikirim ke mikrokontroler utama. *Logic Converter* merupakan rangkaian tambahan yang mengubah tegangan logika serial mini komputer yaitu 1.8 volt menjadi tegangan logika serial mikrokontroler yaitu 3.3 V. Skema rangkaian logic converter dapat dilihat pada Gambar 4.7.



Gambar 4.7 Skematik rangkaian *logic converter*.

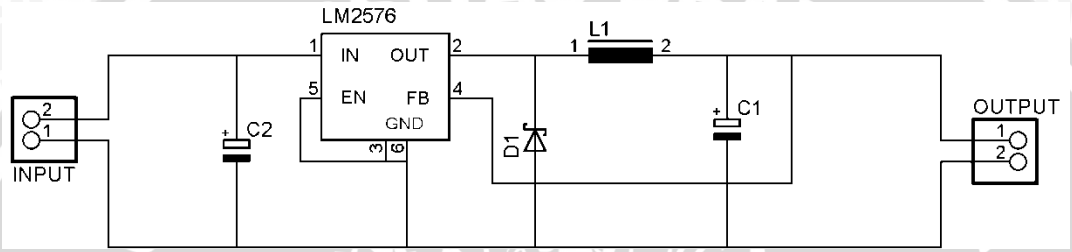
TX_LV merupakan terhubung dengan pin serial mini komputer sedangkan pin TX_HV terhubung dengan pin serial mikrokontroler. LV terhubung dengan tegangan 1.8 V sedangkan HV terhubung dengan tegangan 3.3 V. BSS138 merupakan N-mosfet yang memiliki kecepatan respon cukup tinggi sehingga bisa digunakan untuk aplikasi *switching* berkecepatan tinggi. Selain itu rangkaian ini memiliki Vgs treshold (Vth) yang cukup kecil yakni 1.5 V.

Cara kerja rangkaian untuk mengirimkan logika 0 dari kiri kiri ke kanan maka input TX_LV diberikan logika 0. Tegangan *source* = 0 V dan tegangan *gate* adalah 1.8 V. Vgs akan bernilai 1.8 V dan lebih besar dari Vth sehingga MOSFET akan aktif sehingga TX_HV juga akan berlogika 0. Saat memberikan logika 1 dari kiri ke kanan,

Vgs = *gate*-TX_LV = 0 V sehingga MOSFET tidak aktif dan tegangan pada TX_HV = HV = 3.3 V.

4.2.2.2 Perancangan Catu Daya 5 V

Alat ini membutuhkan catu daya 5V untuk mencatu mikrokontroler dan *rotary encoder*. Sedangkan sumber catu daya utama berupa baterai li-po 11.3 V. Maka dari itu dibutuhkan suatu rangkaian untuk menurunkan tegangan menjadi 5V. Pada perancangan ini digunakan sebuah IC regulator *simple switcher* LM2576 dengan spesifikasi keluaran tegangan 5V dan arus maksimal 3A. Rangkaian regulator tegangan LM2576 ditunjukkan dalam Gambar *4.8*.



Gambar 4.8 Skema rangkaian catu daya sistem.

Rangkaian regulator tegangan dengan IC LM2576 membutuhkan komponen tambahan berupa kapasitor, induktor, dan dioda. Dioda D_1 adalah dioda tipe 1N5822 yang merupakan dioda *schottky*. Kapasitor C_1 bernilai 1000 µF sedangkan kapasitor C_2 bernilai 100 µF. Sedangkan Induktor L_1 memiliki nilai 100 µH. Rangkaian dan spesifikasi komponen didapatkan dari datasheet LM2576.

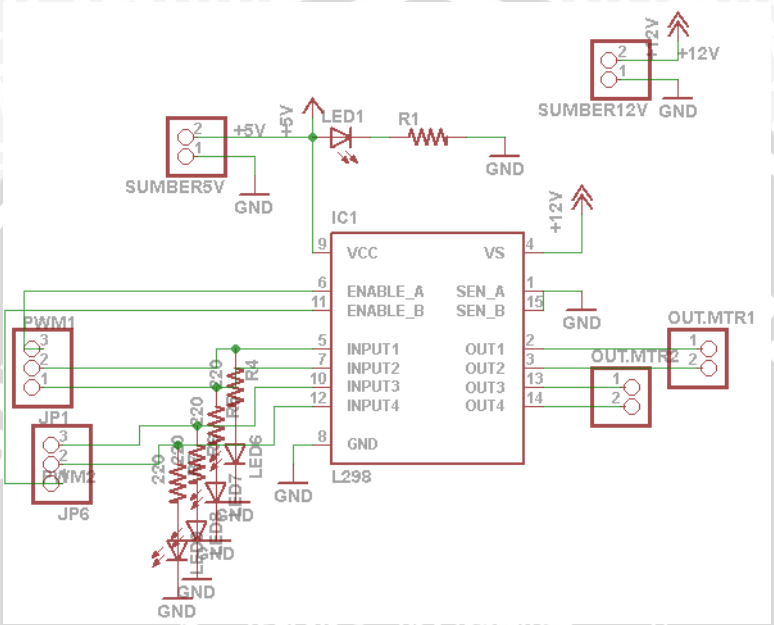
Regulator memiliki keluaran berupa sinyal kotak yang memiliki duty cycle atau t_{on} dan t_{off} yang dikontrol langsung berdasarkan besar tegangan umpan balik dari output. Kapasitor C_2 berfungsi untuk mengurangi *ripple* pada tegangan input. Dioda yang digunakan adalah dioda *schottky* karena memiliki respon yang sangat tinggi dibandingkan dioda biasa. Induktor digunakan untuk meningkatkan efisiensi rangkaian. Kapasitor C_1 digunakan untuk mengubah sinyal keluaran menjadi linear.

4.2.2.3 Perancangan Driver Motor DC

Rangkaian *driver* motor DC berfungsi untuk mengubah sinyal PWM dari modul mikrokontroler dengan tegangan 3.3 V menjadi 12 V, serta mengontrol pergerakan motor. *Driver* motor DC disusun menggunakan IC l298N. L298 aadalah *driver* motor berbasis H-Bridge, mampu menangani beban hingga 4A pada tegangan 6V – 46V. Dalam chip

terdapat dua rangkaian H-Bridge. Selain itu *driver* ini mampu mengendalikan 2 motor sekaligus dengan arus beban 2 A.

Drivel dilengkapi dengan 5 indikator LED. Indikator LED yang pertama berfungsi sebagai penanda adanya sumber tegangan 5 V dari catu daya yang merupakan masukkan pin VCC dari driver. Sedangkan 4 indikator yang lain berfungsi sebagai penanda sinyal keluaran dari mikrokontroler untuk mengatur arah gerak motor.



Gambar 4.9 Skematik rangkaian driver motor l298n.

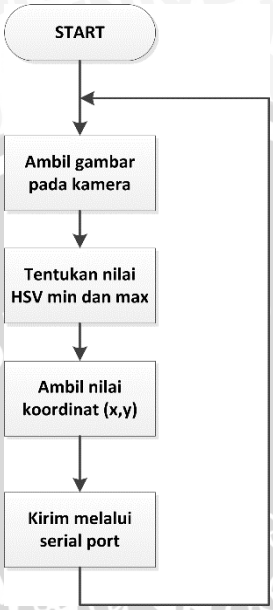
4.3 Perancangan Perangkat Lunak

4.3.1 Pembacaan Posisi Benda Pada Kamera

Pada pemrosesan data kamera pada kamera yang digunakan pada perancangan, kontroler akan melakukan seleksi warna pada kamera dengan menentukan nilai *treshold* dalam bentuk nilai *Hue*, *Saturation*, dan *Value* (HSV). Nilai *treshold* digunakan untuk membedakan warna zona yang dideteksi dengan warna zona pada lapangan yang lain. Kemudian OpenCV akan menentukan titik tengah koordinat dari warna yang terdeteksi. Berdasarkan titik tengah ini maka dapat ditentukan koordinat posisi warna pada layar dalam bentuk koordinat (x,y).

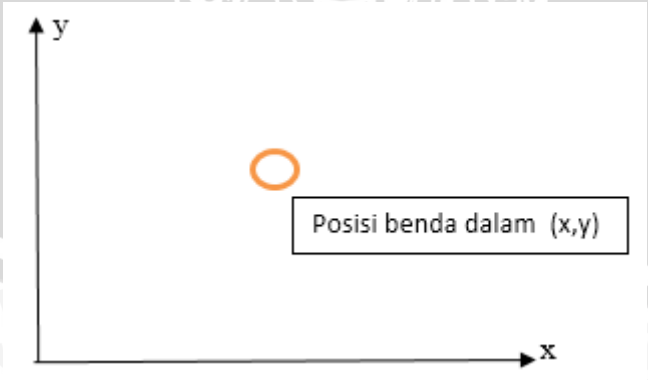
Kamera yang digunakan memiliki resolusi sebesar 720p. Sehingga nilai x maksimal yang bisa diperoleh dari data kamera adalah 1080 dan nilai y maksimal adalah 720. Resolusi maksimal diubah menjadi lebih kecil lagi pada program mini komputer untuk mempercepat proses komputasi dikarenakan pemrosesan data kamera membutuhkan memori yang besar dan proses yang semakin lama. Proses yang lama salah

satunya disebabkan untuk mengalokasikan penyimpanan dan pengambilan data pada memori mini komputer. Nilai maksimal x diubah menjadi 640 sedangkan nilai maksimal y diubah menjadi 480. Sehingga koordinat x memiliki range antara 0-639 dan koordinat y memiliki range 0-479 yang berdasarkan besar resolusi yang ditentukan pada program.



Gambar 4.10 Algoritma membaca data gambar pada kamera.

Data posisi koordinat x,y dari kamera akan dikirim oleh mini PC sebagai kontroler melalui port serial. Pengiriman data dikirim menggunakan format x atau y|Data|CR. Mengirim x atau y sebagai data awal berfungsi sebagai penanda bahwa data yang dikirim adalah nilai koordinat x atau koordinat y, sedangkan CR berfungsi sebagai penanda bahwa data telah selesai dikirim. Sistem pemetaan posisi benda pada layar kamera dapat dilihat pada Gambar 4.11.



Gambar 4.11 Sistem pemetaan pada pembacaan posisi pada layar kamera.

4.3.2 Pembacaan Data Sensor Warna

Sensor warna yang digunakan pada robot adalah sensor warna TCS3200 dengan 1 keluaran dan dilengkapi dengan 4 fotodiode yang dapat diaktifkan secara bergantian.

Keluaran sensor warna dapat dikendalikan dengan mengatur masukan sensor pada pin S_2 dan S_3. Fungsi dari setiap pin masukan pada sensor warna dapat dilihat pada Tabel 4.1. S_0 dan S_1 pada sensor warna diberikan logika 1 agar sensor warna memberikan keluaran pulsa pada frekuensi tinggi.

Tabel 4.1 Tabel fungsi masukkan S_2 dan S_3 pada sensor warna

S_2	S_3	Tipe Photodiode
L	L	Red
L	H	Blue
H	L	Clear (no filter)
H	H	Green

Gambar 4.12 Flowchart algoritma pembacaan warna.

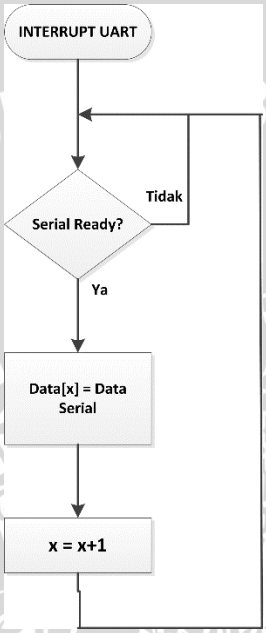
Dengan manfaatkan keluaran dari mikrokontroler untuk memberi masukan logika 0 atau 1 pada masukan S_2 dan S_3 pada sensor warna, maka keluaran dari sensor warna dapat diatur secara bergantian. Algoritma untuk pembacaan data keluaran sensor warna dapat dilihat pada diagram alir pada Gambar 4.12. Proses sampling masukkan dari sensor warna modul mikrokontroler menggunakan interupsi timer. Sensor memiliki 2 keluaran sehingga data disampling pada 2 port input mikrokontroler. Data dari sensor merupakan pulsa digital, sehingga parameter waktu perubahan logika dari pin S_2 dan S_3 sensor akan digunakan untuk menetukan warna yang dideteksi oleh sensor warna.

4.3.3 Algoritma Untuk Mengukur Jarak dan Mengetahui Sudut Hadap

Dalam pembuatan alat ini, robot harus mampu mengetahui jarak dimana robot berdiri dengan jarak warna zona lapangan yang dideteksi. Begitu robot mendeteksi warna

zona lapangan maka servo kepala robot akan menyesuaikan posisi hingga pada layar tangkap kamera posisi zona benar-benar berada di tengah koordinat. Sudut servo kepala akan ditambah atau dikurang sedikit demi sedikit tergantung posisi warna terdeteksi pada layar tangkap kamera.

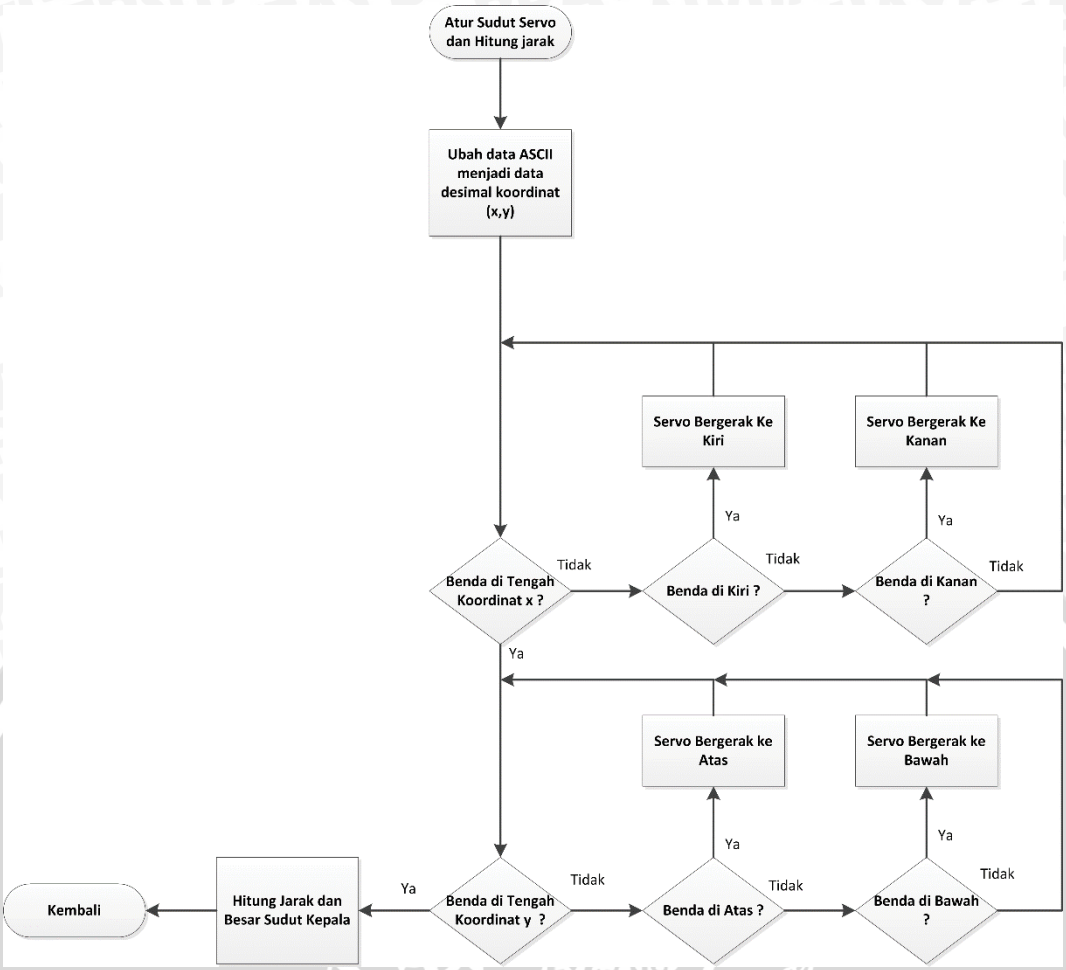
Pertama-tama data ASCII dari mini PC sebagai kontroler kamera diterima melalui interupsi komunikasi serial UART pada modul mikrokontroler. Data awal pada string dari serial berfungsi sebagai penanda bahwa data diterima merupakan koordinat x atau y. Data diterima secara bergantian oleh mikrokontroler seperti diagram alir pada Gambar 4.13.



Gambar 4.13 Diagram alir program membaca data serial pada main kontroler.

Setelah data diterima, data ASCII diubah menjadi data desimal dalam bentuk nilai koordinat x dan y. Mikrokontroler akan menggerakkan servo serial hingga nilai x dan y berada ditengah nilai maksimum atau warna yang dideteksi berada di tengah. Nilai z dan y memiliki nilai maksimum secara berurutan yaitu 640 dan 240 dengan range nilai tengah dari x adalah 300 – 340 dan nilai tengah dari y adalah 230 – 250. Besar sudut servo saat ini kemudian dimasukkan dalam rumus trigonometri untuk menentukan jarak robot dari zona yang dideteksi dan besar sudut hadap.

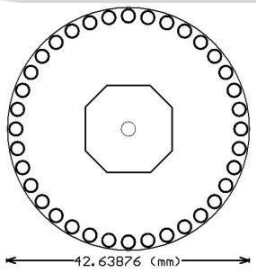
Sudut awal dari kedua aktuator di bagian kepala robot adalah 45°. Perhitungan jarak dilakukan berdasarkan besar tinggi robot dan besar sudut aktuator. Kedua nilai tersebut akan dimasukkan ke persamaan trigonometri yang nantinya akan didapatkan jarak robot dari posisi awal ke jarak zona yang telah dideteksi oleh kamera.



Gambar 4.14 Diagram alir algoritma untuk menentukan jarak dan sudut hadap robot dari zona.

4.3.4 Algoritma Keseluruhan

Pada algoritma keseluruhan robot harus mampu mengukur jarak tempuh agar dapat bergerak menuju zona yang dideteksi oleh kamera. Agar robot dapat melakukan hal ini, umpan balik dari sensor *rotary encoder* akan diperlukan. Sensor rotary encoder yang digunakan adalah sebuah sensor piringan yang memiliki 36 lubang di setiap tepinya. Lubang piringan derajat akan dideteksi oleh sensor optokopler untuk mengetahui jumlah putaran pada roda. Bentuk dan ukuran piringan derajat yang digunakan dapat dilihat pada Gambar 4.15.



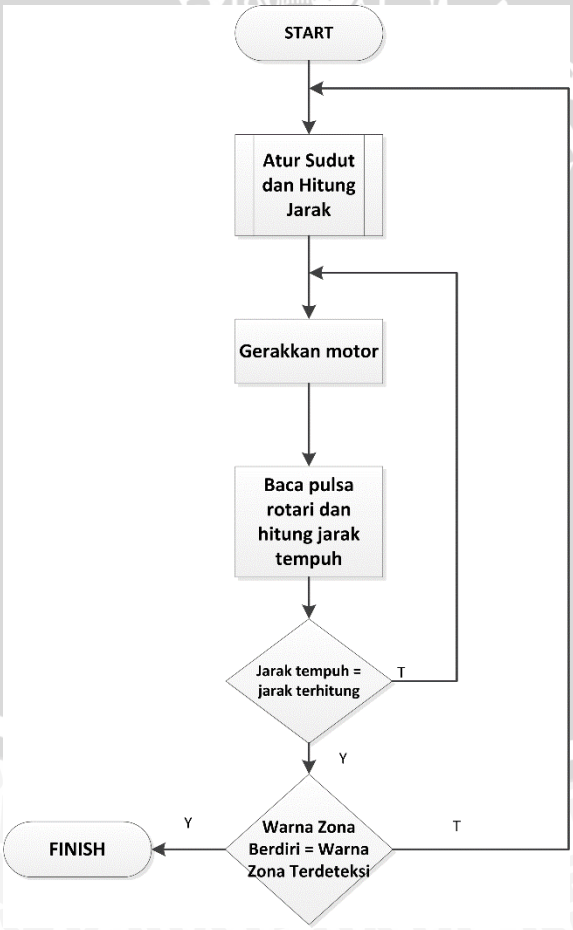
Gambar 4.15 Piringan derajat rotari.

Dengan 36 lubang pada piringan derajat, maka pergerakan roda memiliki ketelitian pembacaan perubahan derajat sebesar 10°. Roda yang digunakan pada alat ini memiliki diameter sebesar 10 cm. Sehingga ketelitian pembacaan jarak dapat dihitung menggunakan persamaan (4-1). Dengan keliling = πd, dimana d adalah diameter roda, maka persamaan (4.1) dapat diubah menjadi persamaan (4-2).

$$Ketelitian\ jarak = \frac{keliling\ roda}{jumlah\ lubang\ pada\ piringan} \quad (4\text{-}1)$$

$$Ketelitian\ jarak = \frac{\pi d}{jumlah\ lubang\ pada\ piringan} \quad (4\text{-}2)$$

Roda memiliki diameter 10 cm dan jumlah lubang yang dimiliki oleh piringan derajat adalah 36 buah, sehingga sensor rotari encoder memiliki ketelitian pembacaan sebesar 8.73 mm. Dengan mengetahui ketelitian pembacaan jarak pada rotari encoder, maka algoritma untuk menghitung jarak tempuh bisa dicari. Diagram alir algoritma untuk menghitung jarak tempuh dapat dilihat pada Gambar 4.16.



Gambar 4.16 Diagram alir algoritma keseluruhan.

Pertama-tama mikrokontroler akan menghitung jarak menggunakan rumus trigonometri sesuai dengan perubahan sudut sendi kepala pada kepala dengan aktuator servo serial dan kamera sebagai sensor warna. Mikrokontroler utama kemudian akan memberikan pulsa PWM untuk menggerakkan motor DC sebagai penggerak roda. Input dari sensor rotari encoder kemudian akan digunakan untuk menentukan jarak yang telah ditempuh oleh robot. Motor akan berhenti saat jarak tempuh dari pembacaan *rotary encoder* telah sama dengan arak terhitung dari rumus trigonometri.

BAB 5

Pengujian dan Analisis

Pengujian dan analisis dilakukan untuk mengetahui apakah sistem telah bekerja sesuai perancangan yang telah dilakukan. Pengujian dilakukan per blok sistem kemudian secara keseluruhan. Adapun pengujian yang dilakukan sebagai berikut:

1) Pengujian sensor kamera
2) Pengujian sensor warna
3) Pengujian *rotary encoder*
4) Pengujian komunikasi serial
5) Pengujian motor servo serial
6) Pengujian pengukuran jarak terdeteksi
7) Pengujian arah hadap robot
8) Pengujian jarak tempuh berdasarkan jarak terhitung
9) Pengujian keseluruhan

5.1 Pengujian Kamera

Pengujian kamera bertujuan untuk mengetahui apakah nilai HSV yang diatur menggunakan program secara real time sesuai dengan nilai HSV yang didapat ketika menggunakan foto dari kamera yang sama. Pengaturan nilai HSV dilakukan dalam program pana mini komputer. Pengujian kamera dilakukan sebanyak 10 kali dengan cara mengambil foto pada warna tiap zona lapangan pada jarak yang berbeda menggunakan sensor kamera kemudian mengujin nilai HSV dari warna pada foto menggunakan software Paint pada komputer. Nilai HSV pada komputer memiliki range sebagai berikut, *Hue* (0-240), *Saturation* (0-240), *Value* (0-240).

Pengujian nilai HSV dilakukan secara real time dengan mengubah nilai HSV pada program sampai warna yang diinginkan bisa terdeteksi. Nilai HSV pada program memiliki lebar data sebesar 8 bit dengan range, yaitu *Hue* (0-255), *Saturation* (0-255), dan *Value* (0-255). Data hasil pengujian merupakan range data tertinggi dan terendah dari 10 kali pengujian yang dilakukan menggunakan kamera dan mini komputer. Data yang diambil merupakan range nilai digital yang diambil pada 10 titik yang berbeda pada gambar dan dapat dilihat pada Tabel 5.1.

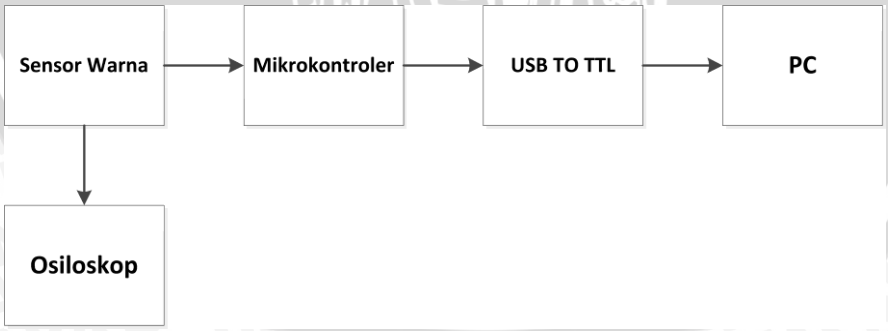
Tabel 5.1 Pengujian nilai HSV pada kamera

Warna	Hue (H)	Sat (S)	Value (V)
Biru Muda	115-130	194-229	130-225
Biru Tua	152-168	133-230	111-188
Hijau	59-87	148-201	47-120

Dari hasil pengujian terhadap nilai HSV dari 3 warna yang akan dideteksi oleh kamera, didapatkan bahwa nilai hue dan saturasi memiliki range yang sempit. Nilai value memiliki range yang lebar karena mudah terpengaruh pencahayaan lingkungan. Sehingga pada pembuatan alat hanya nilai hue dan saturasi yang akan digunakan sebagai pedoman warna.

5.2 Pengujian Sensor Warna

Pengujian sensor warna bertujuan untuk mengetahui apakah keluaran sensor memiliki perubahan yang dapat dilihat dengan kondisi warna yang berbeda. Pengujian dilakukan dengan meletakkan sensor warna pada jarak 1 cm di atas tanah. Hal ini dilakukan untuk membuat kondisi yang semirip mungkin dengan robot humanoid KRSI dimana sensor warna harus diletakkan di kaki sehingga jarak sensor dengan tanah harus < 5cm. Sensor diuji pada 4 warna yang berbeda yaitu biru tua, biru muda, putih, dan hijau sesuai dengan warna yang ada pada lapangan KRSI. Blok diagram pengujian sensor warna dapat dilihat pada Gambar 5.1.



Gambar 5.1 Blok diagram pengujian sensor warna.

Keluaran sensor warna berupa pulsa digital yang berubah-ubah frekuensinya tergantung warna yang dideteksi. Pulsa digital kemudian diamati menggunakan osiloskop dan diamati frekuensi pulsa tegangannya dan dibuktikan dengan menghubungkan pin keluaran sensor dengan pin input mikrokontroler. Mikrokontroler kemudian menghitung

frekuensi sinyal keluaran sensor dan ditampilkan pada serial monitor pada PC dan besar frekuensi dari sensor warna untuk setiap fotodioda yang aktif dapat dilihat pada Tabel *5.2*. Pengujian dilakukan dengan memberi logika 1 pada kedua masukan S_0 dan S_1 pada sensor warna untuk memperbesar frekuensi sensor warna menjadi maksimum.

Gambar 5.2 Sinyal keluaran sensor warna pada osiloskop.

Tabel 5.2 Hasil pengujian frekuensi keluaran sensor warna

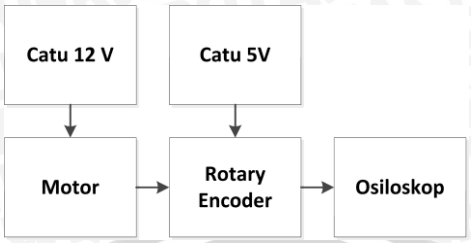
Warna	Frekuensi Keluaran Sensor Warna			
	Clear (kHz)	Merah (kHz)	Hijau (kHz)	Biru (kHz)
Biru Muda	5,370	2,970	4,663	9,536
Biru Tua	2,335	1,906	2,793	3,272
Hijau	2,104	2,275	4,048	3,904
Putih	21,05	9,075	12,94	16,73

Fotodioda dari sensor diaktifkan secara bergantian untuk melihat besar frekuensi keluaran dari sensor warna berdasarkan warna yang dideteksi. Dari hasil pengujian terhadap sensor warna, didapatkan frekuensi keluaran output semakin besar jika warna fotodioda dan LED yang aktif pada sensor warna sama dengan warna yang dideteksi. Gambar 5.2 merupakan hasil pengujian keluaran sensor warna saat S_2 dan S_3 pada input sensor diberikan logika 1 secara bergantian untuk mengaktifkan fotodioda dan LED pada sensor secara bergantian menggunakan program mikrokontroler. Dari hasil pengujian didapatkan frekuensi yang tidak konstan untuk setiap fotodioda yang aktif, perbedaan frekuensi ini digunakan sebagai masukan mikrokontroler untuk menentukan perbedaan warna.

5.3 Pengujian *Rotary Encoder*

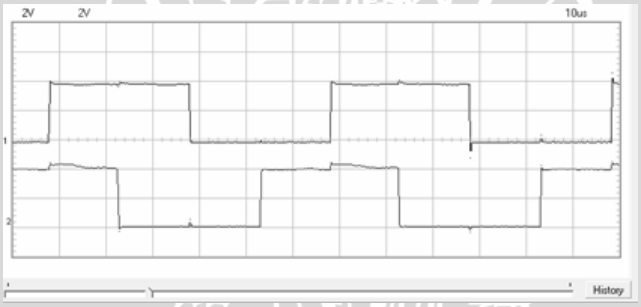
Pengujian ini bertujuan untuk mengetahui keberhasilan sensor *rotary encoder* dalam mendeteksi putaran. Selain itu juga bertujuan untuk menguji keberhasilan sensor dalam mengeluarkan sinyal yang nantinya digunakan untuk mendeteksi jumlah putaran. Peralatan yang digunakan antara lain osiloskop dan kabel probe. Untuk memutar sensor *rotary encoder* supaya berputar dengan konstan maka pada pengujian ini menggunakan

motor DC sebagai pemutarnya. Selain itu juga dibutuhkan catu daya 12 V untuk mencatu motor dan catu daya 5 V untuk mencatu *rotary encoder*.



Gambar 5.3 Blok diagram pengujian *rotary encoder*.

Pengujian dilakukan dengan merangkai peralatan seperti blok diagram dalam Gambar 5.3 Blok diagram pengujian *rotary encoder*.Langkah pertama adalah memasang poros motor pada lubang *rotary encoder* sehingga saat motor berputar *rotary encoder* dapat berputar. Setelah motor dan *rotary encoder* terpasang keluaran rotary encoder disambung ke masukan osiloskop. Setelah semua terpasang, sistem dinyalakan dengan mencatu *rotary encoder* dengan catu 5V dan mencatu motor dengan catu 12 V. *Rotary encoder* dikondisikan berputar searah jarum jam dan kemudian sinyal pada osiloskop diamati hasilnya. Setelah didapatkan sinyal keluaran *rotary encoder* ketika berputar searah jarum jam selanjutnya *rotary encoder* dikondisikan berputar berlawanan arah jarum jam kemudian diamati sinyal keluaranya dengan menggunakan osiloskop.

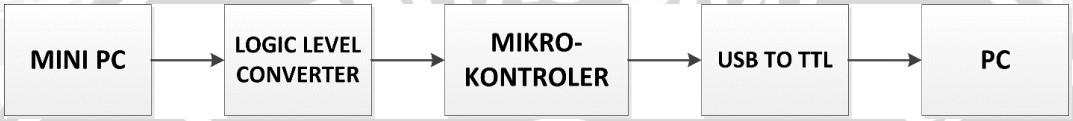


Gambar 5.4 Sinyal pengujian rotary encoder yang berputar dan dikopel oleh motor DC.

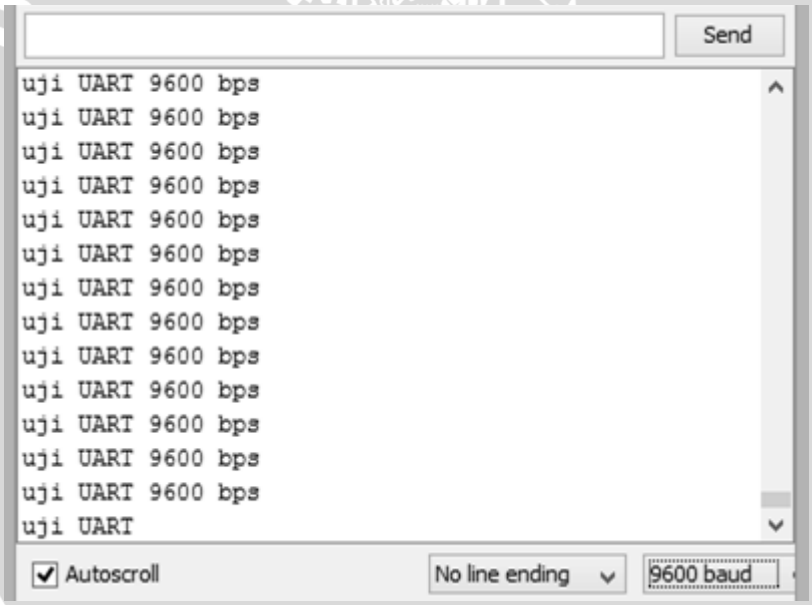
Berdasarkan hasil pengujian pada saat *rotary encoder* diputar searah jarum jam didapatkan hasil sinyal seperti ditunjukkan dalam Gambar 5.4 dimana sinyal yang atas merupakan input chanel A dan yang bawah merupakan input chanel B osiloskop. Input chanel A merupakan output dari *rotary encoder* roda kiri sedangkan input chanel B merupakan output dari rotary encoder roda kanan. Dari hasil pengujian pembacaan putaran didapatkan sinyal dengan amplitudo masing-masing sinyal 5V sehingga keluaran *rotary encoder* bisa langsung dimasukkan ke pin mikrokontroler.

5.4 Pengujian Komunikasi Serial

Pengujian dilakukan untuk mengetahui apakah data serial dapat dikirim dan diterima dengan baik oleh masing-masing perangkat pada alat yang dibuat. Data serial dikeluarkan oleh mikrokontroler melalui komunikasi UART. Pengujian dilakukan dengan cara mengirim data serial "uji UART 9600 bps". Data dikirim pertama kali oleh Mini PC dengan level tegangan data serial 1.8 V. Level tegangan kemudian diubah menjadi 5 V melalui logic level converter. Data serial kemudian diterima oleh mikrokontroler untuk dikirim kembali dan ditampilkan pada serial monitor PC seperti pada Gambar 5.5.



Gambar 5.5 Blok diagram pengujian komunikasi serial.



Gambar 5.6 Data pengujian komunikasi serial pada serial terminal Arduino IDE.

Berdasarkan hasil pengujian dalam Gambar 5.6 terlihat bahwa mikrokontroler dapat mengirimkan data dengan baik dengan tanpa ada karakter yang hilang. Komunikasi serial akan digunakan untuk melakukan pengiriman data hasil proses mini PC ke mikrokontroler utama. Selain itu, komunikasi UART juga akan digunakan untuk menampilkan data hasil perhitungan pada kontroler utama pada PC dengan *baud rate* 9600 bps.

5.5 Pengujian Servo Serial *Dynamixel*

Pengujian servo serial *dynamixel* bertujuan untuk mengetahui keluaran sudut motor servo serial dari nilai data serial yang diberikan. Data serial dikeluarkan oleh microcontroller seperti pada percobaan sebelumnya, tetapi pada percobaan ini, tidak menampilkan data serial dari mikrokontroler melainkan mengamati perubahan besar sudut pada servo dynamixel sesuai data nilai data serial yang diberikan. Servo *dynamixel* memiliki *range* sudut antara 0° - 300° dengan data serial yang dapat diberikan untuk mengubah nilai sudut servo memiliki range antara 0 - 1023. Besar perubahan sudut servo dapat dihitung seperti pada persamaan (5-1).

$$Besar\ Sudut = \frac{Nilai\ Data\ Serial}{1023} \times 300° \quad (5\text{-}1)$$

Untuk Pengujian kali ini, servo serial dipasangi busur lingkaran penuh, dan di bagian horn dipasang plat tipis sebagai jarum penunjuknya. Pemasangan busur pada servo dilakukan dengan menyesuaikan posisi 90° yang diwakili oleh nilai data serial 512. Besar sudut hasil percobaan kemudian dibandingkan dengan besar sudut teori berdasarkan nilai data serial seperti pada Tabel 5.3.

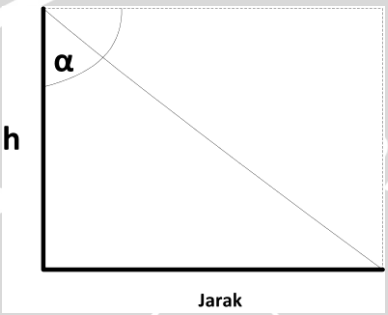
Tabel 5.3 Tabel hasil pengujian servo serial

Nilai Data Serial	Sudut Teori (°)	Sudut (°)	Persen Error (%)
0	0	0	0
100	29.32551	29.5	0.591481
200	58.65103	59	0.591481
300	87.97654	88	0.02666
400	117.3021	118	0.591481
500	146.6276	147	0.253356
600	175.9531	175.5	0.25816
700	205.2786	205	0.1359
800	234.6041	234	0.25816
900	263.9296	263.5	0.16304
1000	293.2551	293	0.08708

Percobaan dilakukan sebanyak 3 kali dan data percobaan yang diambil adalah nilai rata-rata. Dari hasil percobaan pada tabel dapat dilihat bahwa nilai sudut perubahan servo berbanding lurus dengan sudut teori dan antara sudut teori dengan sudut sebenarnya memiliki error < 0.6%. Persen error terbesar perubahan sudut servo dari sudut teori sebesar 0,59%, sedangkan yang terkecil adalah 0% pada sudut 0°.

5.6 Pengujian Pengukuran Jarak benda Menggunakan Trigonometri

Pengujian ini bertujuan untuk membandingkan jarak terhitung dari robot ke benda pada warna yang dideteksi dengan jarak sebenarnya. Perhitungan jarak menggunakan trigonometri seperti pada persamaan (5-2) dapat dilihat pada Gambar 5.7. Besar perubahan sudut pada sendi kepala dan besar tinggi pada kamera pada kepala robot dari permukaan tanah digunakan pada persamaan untuk menetukan jarak robot dari warna yang dideteksi.



Gambar 5.7 Gambar ilustrasi sistem trigonometri.

$$Jarak = \frac{h}{tan(90-\alpha)} \quad (5\text{-}2)$$

Perubahan sudut diukur menggunakan busur derajat penuh yang dipasang di sebelah aktuator. Sedangkan jarak horizontal dan tinggi robot (h) diukur menggunakan penggaris. Jarak horizontal diukur dari titik tengah posisi kamera hingga titik tengah benda yang dideteksi. Sudut α merupakan sudut yang terukur pada busur derajat. Pengujian dilakukan sebanyak 3 kali dan diambil nilai rata-rata sebagai data jarak terhitung. Data hasil percobaan dapat dilihat pada Tabel 5.4.

Tabel 5.4 Hasil Pengujian Pengukuran Jarak Berdasarkan Perubahan Besar Sudut Sendi Kepala

Perubahan Sudut (°)	Jarak Terhitung (cm)	Jarak Sebenarnya (cm)	Persen Error (%)
20	142.8688258	141.75	0.7893
25	111.5143599	110.95	0.50866
30	90.06664199	88.65	1.59802
35	74.26369635	75.1	1.113587
40	61.97118681	62	0.046473
45	52	51.45	1.069
50	43.63318082	42.35	3.02994
55	36.41079199	34.3	6.15391
60	30.022214	30.35	1.08002
65	24.24799822	22.05	9.96825
70	18.92645218	16.45	15.0544

Dari data di atas dapat dilihat jika error hasil perhitungan pada jarak yang dekat semakin besar yaitu pada jarak 16.45 cm dengan kesalahan sebesar 2.37 cm dan persen error sebesar 15,05%. Hal ini dikarenakan benda yang dideteksi semakin besar dan koordinat (x,y) dari gambar yang dideteksi terletak di ujung benda sehingga jarak terukur bisa bertambah sesuai dengan jari-jari benda yang diukur. Zona terkecil dari setiap zona pada lapangan perlombaan KRSI memiliki lebar sebesar 200 mm. Sehingga robot mampu membaca jarak tempuh dengan kesalahan terbesar lebih kecil dari toleransi pembacaan yaitu 200 mm.

5.7 Pengujian Arah Hadap Robot

Pengujian arah hadap pada robot dilakukan untuk mengetahui apakah robot mampu mendeteksi warna memanfaatkan kamera dan mengetahui sudut hadap warna tersebut. Benda yang digunakan sebagai pengujian adalah bola dengan warna hijau. Pengujian dilakukan dengan melakukan meletakkan benda dengan warna biru pada sudut yang berbeda-beda dari arah hadap robot. Hasil pembacaan sudut hadap dapat dilihat pada Tabel 5.5.

Tabel 5.5 Hasil pengujian pembacaan sudut hadap

Sudut Sebenarnya (°)	Sudut terbaca (°)	Persen Error (%)
10	9.3	7
30	27.5	8.333333333
50	51.3	2.6
70	70.6	0.857142857
90	88.5	1.666666667
110	111	0.909090909
130	132.5	1.923076923
150	153.2	2.133333333
170	169.6	0.235294118

Dari hasil pengujian didapatkan kesalahan pembacaan sudut hadap terbesar terdapat pada pembacaan saat benda terdeteksi berada pada sudut 30° dengan persen error sebesar 8.33 %. Hal ini dikarenakan semakin dekat jarak robot dengan warna yang dideteksi maka gambar yang ditangkap kamera semakin mudah terpengaruh noise akibat pengaruh pencahayaan ruang, sehingga menyebabkan sudut tangkap kamera menjadi berubah. Kesalahan pembacaan terkecil terdapat pada sudut sudut 170° dengan persen error 0.23 %.

5.8 Pengujian Jarak Tempuh Robot Berdasarkan Jarak Terhitung

Pengujian ini dilakukan untuk mengetahui apakah robot mampu mengukur perubahan jarak ketika motor bergerak dengan memanfaatkan sensor rotari encoder. Sensor *rotary encoder* yang digunakan memiliki ketelitian pembacaan sebesar 8.73 mm. Pada penelitian ini robot tidak bergerak sehingga perhitungan jarak tempuh hanya berdasarkan putaran roda dan pembacaan sensor putaran.

Pengujian dilakukan dengan memberikan masukan sebagai jarak warna terdeteksi yang nantinya akan dihitung oleh mikrokontroler. Mikrokontroler kemudian akan membaca masukan dari rotari encoder dan mengubahnya menjadi jarak tempuh dan menghentikan pergerakan motor DC saat jarak masukan sama dengan atau lebih dari jarak tempuh terhitung. Hasil pengujian perhitungan jarak tempuh dapat dilihat pada Tabel 5.6 dengan n_Rot Kiri adalah jumlah lubang yang melewati optokopler pada *rotary encoder* kiri dan n_Rot Kanan merupakan jumlah lubang yang melewati optokopler pada *rotary encoder* kanan.

Tabel 5.6 Hasil pengujian jarak tempuh

Jarak Masukkan (mm)	n_Rot Kiri	n_Rot Kan	Jarak Tempuh (mm)	Persen Error (%)
100	11	12	113.49	13.49
200	22	24	218.253	9.1265
300	32	34	305.555	1.851666667
400	41	47	419.047	4.76175
500	57	58	515.079	3.0158
600	67	69	611.111	1.851833333
700	75	82	724.603	3.514714286
800	86	93	820.639	2.579875
900	104	106	934.127	3.791888889
1000	115	116	1021.42	2.142

Berdasarkan hasil percobaan, didapatkan hasil pengukuran dengan kesalahan terbesar pada jarak masukan sebesar 100 mm dengan kesalahan pembacaan sebesar 13.49 mm dan persen error 13.49 %. Sehingga robot mampu membaca jarak tempuh dengan kesalahan terbesar lebih kecil dari toleransi pembacaan yaitu 200 mm. Kesalahan pengukuran disebabkan oleh rendahnya resolusi sensor dan kemampuan pengereman motor yang kurang baik.

5.9 Pengujian Keseluruhan

Pengujian keseluruhan dilakukan dengan menguji kinerja robot pada lapangan perlombaan KRSI. Pengujian bertujuan untuk mengetahui apakah seluruh komponen pendukung robot dapat bekerja sebagaimana mestinya. Ada beberapa parameter yang menentukan keberhasilan kerja robot. Parameter yang pertama adalah robot mampu mengukur jarak dari zona awal ke zonah tengah kemudian ke zona akhir. Parameter kedua adalah robot mampu menghitung jarak tempuh saat bergerak sesuai dengan hasil perhitungan jarak zona terdeteksi. Parameter ketiga adalah robot mampu membaca sudut hadap pada saat arah hadap robot diubah. Kemudian parameter terakhir motor DC sebagai penggerak roda harus berhenti pada setiap perubahan zona lapangan. Pada pengujian robot diangkat untuk berpindah ke zona selanjutnya.

Pengujian dilakukan sebanyak 3 kali dengan pengujian setiap zona dilakukan secara bergantian. Selisih jarak tempuh yang diukur merupakan selisih jarak robot ke zona yang terdeteksi dengan jarak tempuh hasil dari masukkan sensor putaran berdasarkan putaran roda. Pengujian dimulai dari zona awal pada lapangan perlombaan hingga zona tutup seperti pada Tabel 5.7.

Tabel 5.7 Hasil Pengujian Keseluruhan

Zona	Pembacaan Warna	Motor	Selisih Jarak Tempuh (mm)
Zona awal	Biru Tua	Berhenti	-
	Biru Tua	Berhenti	-
	Biru Tua	Berhenti	-
Zona Pambuka	Biru Muda	Berhenti	-
	Biru Muda	Berhenti	-
	Biru Muda	Berhenti	-
Zona Tengah	Putih	Berhenti	33.5 mm
	Putih	Berhenti	26.7 mm
	Putih	Berhenti	31 mm
Zona Bambangan	Biru Tua	Berhenti	-
	Biru Tua	Berhenti	-
	Biru Tua	Berhenti	-
Zona Perang	Hijau	Berhenti	-
	Hijau	Berhenti	-
	Hijau	Berhenti	-
Zona Tutup	Biru Tua	Berhenti	65.4 mm
	Biru Tua	Berhenti	60.5 mm
	Biru Tua	Berhenti	66.3 mm

Berdasarkan pengujian didapatkan kesalahan jarak tempuh terbesar adalah 65,4 mm dan motor mampu berhenti di setiap perubahan zona. Dari hasil percobaan dapat dilihat bahwa sensor warna mampu membaca setiap warna zona dengan benar dan robot mampu mengukur jarak tempuh dengan kesalahan lebih kecil dari toleransi sebesar 200 mm. Sehingga dapat disimpulkan seluruh komponen pendukung kerja robot mampu bekerja dengan baik.

BAB VI

Kesimpulan dan Saran

6.1 Kesimpulan

Berdasarkan hasil percobaan, dapat ditarik kesimpulan sebagai berikut.

1. Robot memanfaatkan perubahan besar sudut sendi kepala sebagai pedoman arah sudut hadap dan jarak. Data dari sensor kamera berupa koordinat x,y dengan resolusi 640 x 480 pada layar kamera.
2. Robot mampu membedakan warna pada setiap zona lapangan perlombaan KRSI memanfaatkan perubahan frekuensi pada keluaran sensor warna dengan tingkat keberhasilan 100%.
3. Robot mampu menghitung jarak dari posisi robot berdiri ke zona yang terdeteksi oleh sensor kamera dengan kesalahan pembacaan tebesar pada jarak 16.45 cm dan perubahan besar sudut sendi kepala sebesar 70° dengan persen error sebesar 15.05 % memanfaatkan rumus trigonometri. Hal ini disebabkan karena kamera yang digunakan mudah terpengaruh pencahayaan ruang. Selain itu, Robot mampu membaca sudut hadap robot berdasarkan perubahan besar sudut sendi kepala pada saat pendeteksian warna dengan kesalahan pembacaan terbesar terjadi pada sudut 30° dengan persen error sebesar 8,33%.
4. Robot mampu melakukan perhitungan jarak tempuh dengan memanfaatkan sensor *rotary encoder* dengan kesalahan perhitungan terbesar sebesar 34.127 mm pada jarak masukan 900 mm.

6.2 Saran

Untuk mendukung penelitian kedepan yang lebih baik dapat dituliskan saran sebagai berikut.

1. Menggunakan algoritma yang lebih baik untuk pergerakan kamera dalam mencari daerah deteksi, seperti PID, dll.
2. Menggunakan rotari encoder dengan ketelitian lebih tinggi dan pemasangan mekanik yang lebih presisi.
3. Menerapkan metode pada robot humanoid.

Daftar Pustaka

Adam Goode, A. R. 2011. *Color-tracking Explanation.* http://www.cmucam.org/projects/cmucam4/wiki/Color-tracking_Explanation (dipetik Mei 2015)

Alciatore, D. G., & Histand, M. B. 2012. *Introduction To Mechatronics and Measurement Systems.* New York: McGraw-Hill.

Atmel. 2007. *8-bit AVR with 16K Bytes In-System Programmable Flash ATMega16/16L.* San Jose: Atmel.

Buana, G. 2013. *Implementasi Logika Fuzzy Sebagai Perintah Gerakan Tari Pada Robot Humanoid KRSI Menggunakan Sensor Kamera CMUCAM4.* Malang: Universitas Brawijaya.

Dikti. 2014. *Panduan KRSI 2015-ver Oktober 2014.* Jakarta Pusat: Dikti.

Hall, R. 2013. *ODROID Magazine* (1st ed.). Hard Kernel.

Itseez. 2014. *The OpenCV Tutorials.* Nizhny Novgorod: Itseez.

Latoschik, M. E. 2005. *Color Models.* Diambil kembali dari Realtime 3D Computer Graphics / Virtual Reality: web.eecs.utk.edu/~huangj/cs456/notes/456_color.pdf

Robotis. 2006. *AX-12/AX-12+/AX-12A - ROBOTIS e-MANUAL.* Robotis.

STMicroelectronics. 2012. *STM32F405xx STM32F407xx Datasheet.* STMicroelectronics.

STMicroelectronics. 2012. *UM1472 User Manual STM32F4Discovery.* STMicroelectronics

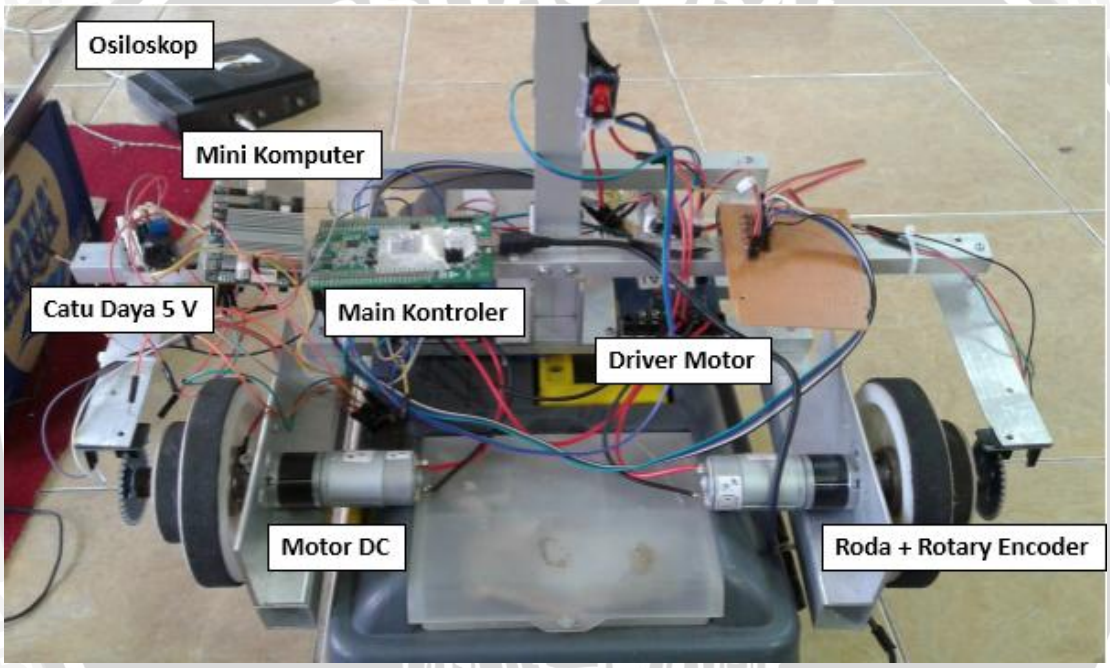
TAOS. 2009. *TCS3200, TCS3210 PROGRAMMABLE COLOR LIGHT-TO-FREQUENCY CONVERTER.* TAOS.

Yilmaz, A., Javed, O., & Shah, M. 2006. Object Tracking : A Survey. *ACM Computing Surveys*, *38*(14), 45.

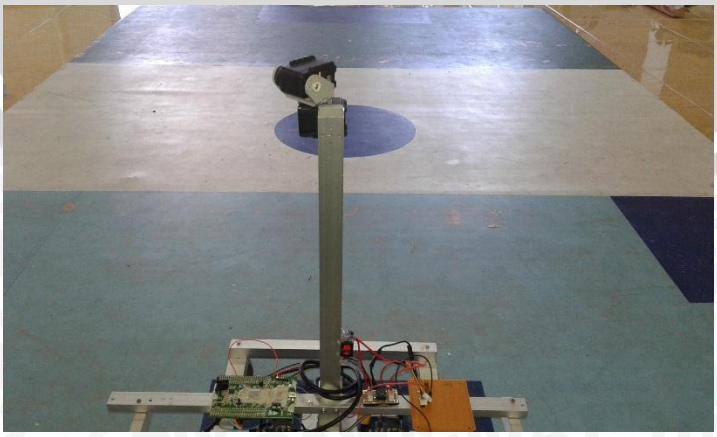
LAMPIRAN 1

FOTO ALAT

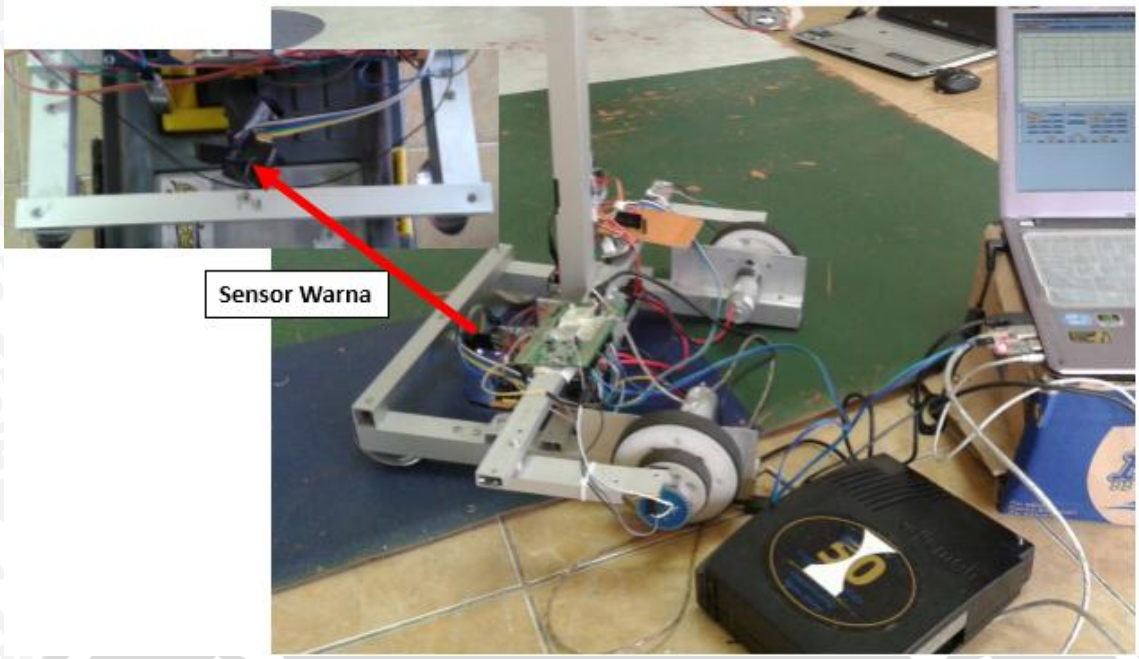
Gambar 1 Kamera Pada Kepala Robot.



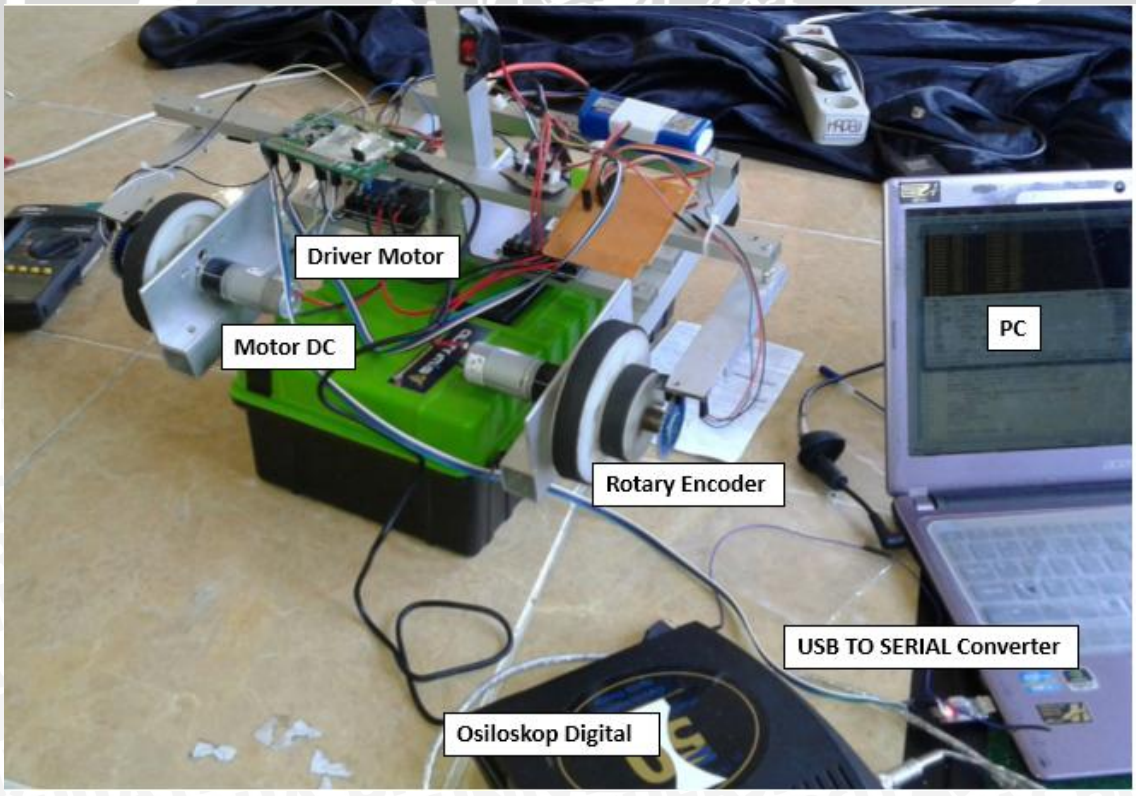
Gambar 2 Foto Alat.



Gambar 3 Pengujian Pada Lapangan KRSI.



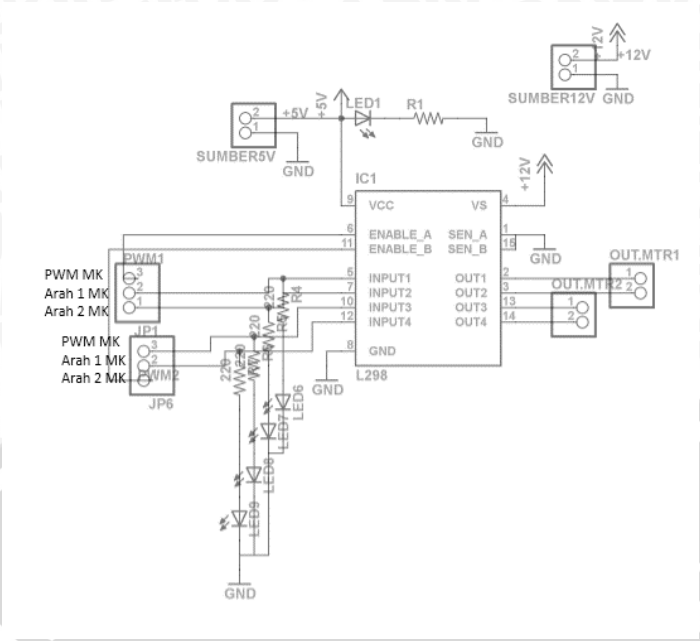
Gambar 4 Pengujian Sensor Warna.



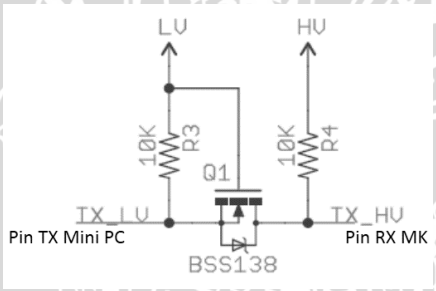
Gambar 5 Pengujian Rotari Encoder.

LAMPIRAN 2

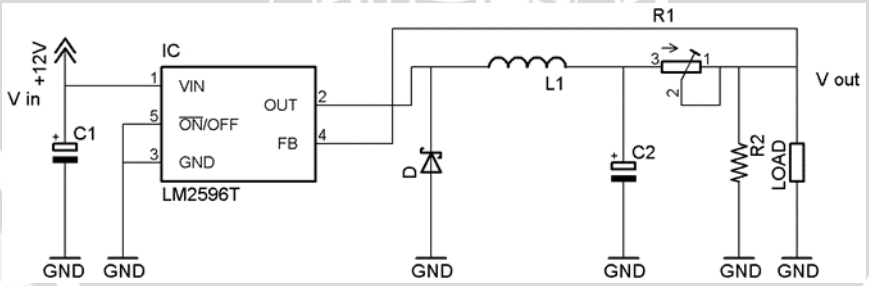
RANGKAIAN KESELURUHAN



Gambar 1 Rangkaian Driver Motor



Gambar 2 Rangkaian Logic Level Converter



Gambar 3 Catu Daya 5 V

LAMPIRAN 3

LISTING PROGRAM

```
#include "stm32f4xx.h"
#include "stm32f4xx_rcc.h"
#include "stm32f4xx_gpio.h"
#include "stm32f4xx_tim.h"
#include "stm32f4xx_exti.h"
#include "stm32f4xx_syscfg.h"
#include "stm32f4xx_usart.h"
#include "misc.h"
#include "delay.h"
#include <stdio.h>
#include <math.h>
#include <string.h>

//=================SENSOR
WARNA================================
=====//
//register kontrol untuk mengendalikan timer 2
#define  t2_on          TIM_Cmd(TIM2, ENABLE)
#define  t2_off         TIM_Cmd(TIM2, DISABLE)
#define     biru_muda 1
#define     putih              2
#define     hijau              3
#define     biru_tua    4
//register pencacah
#define  reset_t2       TIM2->CNT=0x00000000
uint32_t CCR1_Val = 80000;
char warna;
float white,red,blue,green;
void TCS3200Mode(unsigned char mode);
void detectColor(void);
void TIM_Init();
void sum_color(void);
float colorRead(unsigned char color,  unsigned char
LEDstate);
//=======================================
============================//

//================MOTOR & ROTARY
ENCODER============================//
#define   PWM_TIM3_PERIODE    0xFF
            // periode   (0xFF => 8bit)
#define   PWM_TIM3_PRESCALE   327
            // prescaler (328 => 1kHz)
#define   PWM_TIM3_POLARITY
TIM_OCPolarity_High
#define   PWM_R TIM3->CCR1
          //CCR=PULSE, MAX=>period=8bit=255
#define   PWM_L TIM3->CCR2
int Rcount=0, Lcount=0;
float jarak;
int in_jarak;
char kata[16];
char rx_data;
unsigned char pwm_max = 60;
uint16_t CCR1_Val = 255; //mengikuti pwm max
float
target_sudut,error=0,PID,error_dif,error_prev,error_su
m;
float last1_error=0;
float last2_error=0;
float P_kiri;
float P_kanan;
float count_kel;

void PID_maju(float cp, float ci, float cd);
void m_cw();
void m_ccw();
void ml_cw();
void ml_ccw();
void stop_kanan();
void stop_kiri();
//=======================================
============================//

//=========================DATA
KAMERA============================//
#define MAX_STRLEN 12

volatile char received_string[MAX_STRLEN+1];
int z,y,x;
char kata[16];
char rx_data,l=1;
int  t=0, data_usart, hasil;
unsigned char data_array[6];
int val;
void convert_data();
void ubah_sudut();
//=======================================
============================//

//=================SERVO SERIAL
DYNAMIXEL=========================//
#define AX_MODEL_NUMBER_L          0
#define AX_MODEL_NUMBER_H          1
#define AX_VERSION                 2
#define AX_ID                      3
#define AX_BAUD_RATE               4
#define AX_RETURN_DELAY_TIME       5
#define AX_CW_ANGLE_LIMIT_L        6
#define AX_CW_ANGLE_LIMIT_H        7
#define AX_CCW_ANGLE_LIMIT_L       8
#define AX_CCW_ANGLE_LIMIT_H       9
#define AX_SYSTEM_DATA2            10
#define AX_LIMIT_TEMPERATURE       11
#define AX_DOWN_LIMIT_VOLTAGE      12
#define AX_UP_LIMIT_VOLTAGE        13
#define AX_MAX_TORQUE_L            14
#define AX_MAX_TORQUE_H            15
#define AX_RETURN_LEVEL            16
#define AX_ALARM_LED               17
#define AX_ALARM_SHUTDOWN          18
#define AX_OPERATING_MODE          19
#define AX_DOWN_CALIBRATION_L      20
#define AX_DOWN_CALIBRATION_H      21
#define AX_UP_CALIBRATION_L        22
#define AX_UP_CALIBRATION_H        23
#define AX_TORQUE_ENABLE           24
#define AX_LED                     25
#define AX_CW_COMPLIANCE_MARGIN    26
#define AX_CCW_COMPLIANCE_MARGIN   27
#define AX_CW_COMPLIANCE_SLOPE     28
#define AX_CCW_COMPLIANCE_SLOPE    29
#define AX_GOAL_POSITION_L         30
#define AX_GOAL_POSITION_H         31
#define AX_GOAL_SPEED_L            32
#define AX_GOAL_SPEED_H            33
#define AX_TORQUE_LIMIT_L          34
#define AX_TORQUE_LIMIT_H          35
#define AX_PRESENT_POSITION_L      36
#define AX_PRESENT_POSITION_H      37
#define AX_PRESENT_SPEED_L         38
#define AX_PRESENT_SPEED_H         39
#define AX_PRESENT_LOAD_L          40
#define AX_PRESENT_LOAD_H          41
#define AX_PRESENT_VOLTAGE         42
#define AX_PRESENT_TEMPERATURE     43
#define AX_REGISTERED_INSTRUCTION  44
#define AX_PAUSE_TIME              45
#define AX_MOVING                  46
#define AX_LOCK                    47
#define AX_PUNCH_L                 48
#define AX_PUNCH_H                 49
#define OFF                0
#define ON                 1
#define LEFT
                           0
```

```
#define RIGTH                   1
#define AX_BYTE_READ            1
#define AX_BYTE_READ_POS        2
#define AX_RESET_LENGTH
        2
#define AX_ACTION_LENGTH
        2
#define AX_ID_LENGTH            4
#define AX_LR_LENGTH            4
#define AX_SRL_LENGTH           4
#define AX_RDT_LENGTH           4
#define AX_LEDALARM_LENGTH      4
#define AX_SALARM_LENGTH        4
#define AX_TL_LENGTH            4
#define AX_VL_LENGTH            6
#define AX_CM_LENGTH            6
#define AX_CS_LENGTH            5
#define AX_CCW_CW_LENGTH        8
#define AX_BD_LENGTH            4
#define AX_TEM_LENGTH           4
#define AX_MOVING_LENGTH        4
#define AX_RWS_LENGTH           4
#define AX_VOLT_LENGTH          4
#define AX_LED_LENGTH           4
#define AX_TORQUE_LENGTH        4
#define AX_POS_LENGTH           4
#define AX_GOAL_LENGTH          5
#define AX_MT_LENGTH            5
#define AX_PUNCH_LENGTH         5
#define AX_SPEED_LENGTH         5
#define AX_GOAL_SP_LENGTH       7
#define AX_ACTION_CHECKSUM
        250
#define BROADCAST_ID            254
#define AX_START                255
#define AX_CCW_AL_L             255
#define AX_CCW_AL_H             3
#define TIME_OUT                10
#define TX_DELAY_TIME
        400
#define Tx_MODE                 1
#define Rx_MODE                 0
#define LOCK                    1
unsigned char Checksum;
unsigned char Direction_Pin;
unsigned char Time_Counter;
unsigned char Incoming_Byte;
unsigned char Position_High_Byte;
unsigned char Position_Low_Byte;
unsigned char Speed_High_Byte;
unsigned char Speed_Low_Byte;
unsigned char Load_High_Byte;
unsigned char Load_Low_Byte;
int moveSpeed(unsigned char ID, int Position, int
Speed);
int nilai_sudut1=512;
int nilai_sudut2=512;
//=======================================
==========================//

GPIO_InitTypeDef GPIO_InitStructure;
EXTI_InitTypeDef EXTI_InitStructure;
NVIC_InitTypeDef NVIC_InitStructure;
USART_InitTypeDef USART_InitStructure;
TIM_TimeBaseInitTypeDef
TIM_TimeBaseStructure;
TIM_OCInitTypeDef       TIM_OCInitStructure;

//Eksternal Interrupt
void EXTI0_IRQHandler(void);
void EXTILine0_Config(void);
void EXTI1_IRQHandler(void);
void EXTILine1_Config(void);
```

```
//USART
void USART3_PutChar(char c);
void usart_puts(char *data);
void usart_inisialisasi();
void USART_put(USART_TypeDef* USARTx,
volatile char *s);

void PIN_Configuration(void);

void USART1_IRQHandler(void)
{
                    if(
USART_GetITStatus(USART1, USART_IT_RXNE)
){
                                static uint8_t cnt = 0;
                                rx_data = USART1-
>DR;
                                if( (rx_data != 'z') &&
(cnt < MAX_STRLEN) ){
        received_string[cnt] = rx_data;

        data_array[t]=received_string[cnt];
                                        cnt++;
                                        t++;
                                }
                                else
                                {

        convert_data();
                                        cnt = 0;

        read_x_y();
                                }
}
}

void EXTI0_IRQHandler(void)
{
 if(EXTI_GetITStatus(EXTI_Line0) != RESET)
 {
        Rcount++;
        count_kel+=8.73015;
        /* Clear the EXTI line 0 pending bit */
        EXTI_ClearITPendingBit(EXTI_Line0);
 }
}

void EXTI1_IRQHandler(void)
{
 if(EXTI_GetITStatus(EXTI_Line1) != RESET)
 {
          Lcount++;
          /* Clear the EXTI line 0 pending bit */
          EXTI_ClearITPendingBit(EXTI_Line1);
  }
}

int main(void)
{
        SystemInit();
        SysTick_Init();
        PIN_Configuration();
  EXTILine0_Config();
  EXTILine1_Config();
        TIM_Init();
        usart_inisialisasi();

        float count_kel=8.73015;

        detect_kamera();
        while(in_jarak > count_kel)
```

```
        {
                PID_maju(18,6,0);
                if(warna==biru_muda)
                {
                        stop_kiri();
                        stop_kanan();
                        delay(5000);
                }

        }
        stop_kiri();
        stop_kanan();
        delay(5000);

        detect_kamera();
        while(in_jarak > count_kel)
        {
                PID_maju(18,6,0);
                if(warna==putih)
                {
                        stop_kiri();
                        stop_kanan();
                        delay(5000);
                }
                if(warna==hijau)
                {
                        stop_kiri();
                        stop_kanan();
                        delay(5000);
                }
        }
        stop_kiri();
        stop_kanan();
        delay(5000);


   while(1)
   {
   }
}

void detect_kamera()
{
        while(x>350)
        movespeed(1,nilai_sudut1-10,1000);
        while(x<290)
        movespeed(1,nilai_sudut1+10,1000);
        while(y>270)
        movespeed(2,nilai_sudut2-10,1000);
        while(y<210)
        movespeed(2,nilai_sudut2+10,1000);
        float sudut2=(nilai_sudut2/1023)*300;
        in_jarak=tan(sudut2)*55.5;

}
void Read_x_y()
{
        switch(data_array[0])
        {
        case 'a':
                x=hasil;

        sprintf(kata,"x=%d\r",x);
                        usart_puts(kata);
                break;
        case 'b':
                y=hasil;
                sprintf(kata,"y=%d\r",y);
                usart_puts(kata);
                break;
        case 'c':
                break;
        }
}

void convert_data()
{
        if (t == 0);

        else if (t == 5)
        {
                data_array[1]=data_array[1]-48;
                data_array[2]=data_array[2]-48;
                data_array[3]=data_array[3]-48;
                data_array[4]=data_array[4]-48;
                hasil = (data_array[1] * 1000) +
(data_array[2] * 100) + (data_array[3] * 10) +
(data_array[4]);
                t = 0;
        }
        else if (t == 4)
        {
                data_array[1] = data_array[1] -
48;
                data_array[2] = data_array[2] -
48;
                data_array[3] = data_array[3] -
48;
                hasil = (data_array[1] * 100) +
(data_array[2] * 10) + (data_array[3]);
                t = 0;
        }
        else if (t == 3)
        {
                data_array[1] = data_array[1] -
48;
                data_array[2] = data_array[2] -
48;
                hasil = (data_array[1] * 10) +
(data_array[2]);
                t = 0;
        }
        else if (t == 2)
        {
                data_array[1] = data_array[1] -
48;
                hasil = (data_array[1]);
                t = 0;
        }
        else t=0;
}

void PID_maju(float cp, float ci, float cd)
{
        error = Lcount - Rcount;     //hitung error
   error_dif = error - error_prev;   //hitung delta error
   error_prev = error;               //masukan error ke
error_prev, pada proses selanjutnya adalah error
sebelumnya
   error_sum += error;               //akumulasikan error
   PID = (cp*error) + (ci*error_dif) + (cd*error_sum);
//PID

   if(PID>0)ml_cw();  //jika hasil PID positif maka arah
motor cw
   else if(PID<0)ml_ccw(); //jika hasil PID negatif
maka arah motor ccw
   else stop_kiri(); //jika hasil PID 0 maka motor diam

   //karena akan dimasukan sebagai ke dalam PWM,
PID harus positif
   PID = fabs(PID);

   //batasi nilai PID dengan nilai pwm_max
   if(PID>pwm_max)PID=pwm_max;
```

```
   //PID sudah siap dimasukan ke PWM
   PWM_L = (unsigned char)PID;
}

void m_cw() //set arah motor kanan
{
        GPIO_SetBits(GPIOD, GPIO_Pin_1);
        // arah_MotorR_N = 1
        GPIO_ResetBits(GPIOD, GPIO_Pin_2);  //
arah_MotorR_S = 0
}

void ml_cw() //set arah motor kiri
{
                GPIO_SetBits(GPIOD,
GPIO_Pin_3);    // arah_MotorL_N = 1
                GPIO_ResetBits(GPIOD,
GPIO_Pin_6);    // arah_MotorL_S = 0
}
void ml_ccw()  //set arah motor kiri
{
        GPIO_ResetBits(GPIOD, GPIO_Pin_3);
        // arah_MotorL_N = 0
        GPIO_SetBits(GPIOD, GPIO_Pin_6);
        // arah_MotorL_S = 1
}

void stop_kanan()  //set diam
{
        GPIO_SetBits(GPIOD, GPIO_Pin_1);
        // arah_MotorR_N = 1
        GPIO_SetBits(GPIOD, GPIO_Pin_2);  //
arah_MotorR_S = 1
}

void stop_kiri()  //set diam
{
        GPIO_SetBits(GPIOD, GPIO_Pin_3);
        // arah_MotorR_N = 1
        GPIO_SetBits(GPIOD, GPIO_Pin_6);  //
arah_MotorR_S = 1
}

void sum_color(void)
{
        unsigned char r=0;
        static float sum_red,sum_blue=0;
        detectColor();
        sprintf(kata," White= %.2f\t Red= %.2f\t
Blue= %.2f\t Green= %.2f\t \r",white,red,blue,green);
        usart_puts(kata);
}
}

void detectColor(void)
{
        white = colorRead(0,1);
        red = colorRead(1,1);
        blue = colorRead(2,1);
        green = colorRead(3,1);
}

float colorRead(unsigned char color,  unsigned char
LEDstate){
        float readPulse;
        //Aktifkan sensor pada frekuensi paling
tinggi
        TCS3200Mode(1);

        //Setting S2 dan S3 untuk memilih fotodioda
yang aktif
        if(color == 0){//white
                GPIO_ResetBits(GPIOE,
GPIO_Pin_3);
                GPIO_SetBits(GPIOE,
GPIO_Pin_4);
                }
        else if(color == 1){//red
                GPIO_ResetBits(GPIOE,
GPIO_Pin_3);
                GPIO_ResetBits(GPIOE,
GPIO_Pin_4);
                }
        else if(color == 2){//blue
                GPIO_SetBits(GPIOE,
GPIO_Pin_3);
                GPIO_ResetBits(GPIOE,
GPIO_Pin_4);
                }
        else if(color == 3){//green
                GPIO_SetBits(GPIOE,
GPIO_Pin_3);
                GPIO_SetBits(GPIOE,
GPIO_Pin_4);
                }
        //start counting saat sinyal high, berhenti
ketika low
        while(GPIO_ReadInputDataBit(GPIOE,
GPIO_Pin_2)==0){} reset_t2;t2_on;
        while(GPIO_ReadInputDataBit(GPIOE,
GPIO_Pin_2)==1){} t2_off;

        readPulse = TIM2->CNT;
        //nilai maksimal read Pulse
        if(readPulse < .1)
        {
                readPulse = 80000;
        }
        return readPulse;
}

void TCS3200Mode(unsigned char mode)
{

  if(mode == 0){
  //power OFF mode
  GPIO_ResetBits(GPIOE, GPIO_Pin_7);
  GPIO_ResetBits(GPIOE, GPIO_Pin_6);
  GPIO_ResetBits(GPIOE, GPIO_Pin_5);
  }
  else if(mode == 1){
  //Perbandingan 1:1, highest sensitivity
  GPIO_SetBits(GPIOE, GPIO_Pin_6);
  GPIO_SetBits(GPIOE, GPIO_Pin_5);
  }
  else if(mode == 2){
  //Perbandingan 1:5
  GPIO_SetBits(GPIOE, GPIO_Pin_6);
  GPIO_ResetBits(GPIOE, GPIO_Pin_5);
  }
  else if(mode == 3){
  //Perbandingan 1:50
  GPIO_ResetBits(GPIOE, GPIO_Pin_6);
  GPIO_SetBits(GPIOE, GPIO_Pin_5);
  }

  return;

}

void PIN_Configuration(void)
{
        //-----------------------Output Setting
S0,S1,S2,S3----------
```

```
	RCC_AHB1PeriphClockCmd
(RCC_AHB1Periph_GPIOE, ENABLE);
	GPIO_InitStructure.GPIO_Pin =
GPIO_Pin_3| GPIO_Pin_4| GPIO_Pin_5| GPIO_Pin_6 |
GPIO_Pin_7;
	GPIO_InitStructure.GPIO_Mode =
GPIO_Mode_OUT;		// Pin ini memiliki Mode
Ouput
	GPIO_InitStructure.GPIO_OType =
GPIO_OType_PP;			// Pin
bersifat Push Pull (Pull-up, down or no Pull)
	GPIO_InitStructure.GPIO_Speed =
GPIO_Speed_100MHz;		//
kecepatan clock(2, 25, 50 or 100 MHz)
	GPIO_InitStructure.GPIO_PuPd =
GPIO_PuPd_DOWN;			// pin
tidak diberikan pull up
	GPIO_Init(GPIOE, &GPIO_InitStructure);
							//
inisialisasi peripheral GPIO sesuai parameter typdef
diatas
	//------------------------Input Pembaca Output
Sensor Warna------------------------
	RCC_AHB1PeriphClockCmd
(RCC_AHB1Periph_GPIOE, ENABLE);
	GPIO_InitStructure.GPIO_Pin =
GPIO_Pin_2;
	GPIO_InitStructure.GPIO_Mode =
GPIO_Mode_IN;
	GPIO_InitStructure.GPIO_OType =
GPIO_OType_PP;
	GPIO_InitStructure.GPIO_Speed =
GPIO_Speed_100MHz;
	GPIO_InitStructure.GPIO_PuPd =
GPIO_PuPd_DOWN;
	GPIO_Init(GPIOE, &GPIO_InitStructure);

	//-------------PIN logika Arah Driver Motor--
----------
	RCC_AHB1PeriphClockCmd
(RCC_AHB1Periph_GPIOD, ENABLE);
	GPIO_InitStructure.GPIO_Pin =
GPIO_Pin_1| GPIO_Pin_2| GPIO_Pin_3| GPIO_Pin_4|
GPIO_Pin_6;
	GPIO_InitStructure.GPIO_Mode =
GPIO_Mode_OUT;		// Pin ini memiliki Mode
Ouput
	GPIO_InitStructure.GPIO_OType =
GPIO_OType_OD;			// Pin
bersifat Open Drain
	GPIO_InitStructure.GPIO_Speed =
GPIO_Speed_100MHz;		//
kecepatan clock(2, 25, 50 or 100 MHz)
	GPIO_InitStructure.GPIO_PuPd =
GPIO_PuPd_NOPULL;		// pin
tidak diberikan pull up
	GPIO_Init(GPIOD, &GPIO_InitStructure);
							//
inisialisasi peripheral GPIO sesuai parameter typdef
diatas

	//-------------PIN Arah Rotary-----------
	RCC_AHB1PeriphClockCmd
(RCC_AHB1Periph_GPIOA, ENABLE);
	GPIO_InitStructure.GPIO_Pin =
GPIO_Pin_2 | GPIO_Pin_3;
	GPIO_InitStructure.GPIO_Mode =
GPIO_Mode_IN;
	GPIO_InitStructure.GPIO_OType =
GPIO_OType_PP;
	GPIO_InitStructure.GPIO_Speed =
GPIO_Speed_100MHz; //kecepatan clock(2, 25, 50 or
100 MHz)
	GPIO_InitStructure.GPIO_PuPd =
GPIO_PuPd_NOPULL;
	GPIO_Init(GPIOA, &GPIO_InitStructure);

	//--------------Output LED Indikator------
	RCC_AHB1PeriphClockCmd(RCC_AHB1
Periph_GPIOD, ENABLE);
	GPIO_InitStructure.GPIO_Pin =
GPIO_Pin_12 | GPIO_Pin_13| GPIO_Pin_14|
GPIO_Pin_15;
	GPIO_InitStructure.GPIO_Mode =
GPIO_Mode_OUT;
	GPIO_InitStructure.GPIO_OType =
GPIO_OType_PP;
	GPIO_InitStructure.GPIO_Speed =
GPIO_Speed_100MHz;
	GPIO_InitStructure.GPIO_PuPd =
GPIO_PuPd_NOPULL;
	GPIO_Init(GPIOD, &GPIO_InitStructure);

	return;
}

void TIM_Init()
{
	//-----------------TIM2----------------------
-
	RCC_APB1PeriphClockCmd(RCC_APB1P
eriph_TIM2,ENABLE);
	IM_TimeBaseInitTypeDef
TIM_TimeBaseStructure;
	//frekuensi timer=
84000000/(83+1)=1000000Hz
	TIM_TimeBaseStructure.TIM_Prescaler =
83;
	TIM_TimeBaseStructure.TIM_CounterMod
e = TIM_CounterMode_Up;
	TIM_TimeBaseStructure.TIM_Period =
CCR1_Val;
	TIM_TimeBaseStructure.TIM_ClockDivisio
n = 0;
	TIM_TimeBaseStructure.TIM_RepetitionCo
unter = 0;
	TIM_TimeBaseInit(TIM2,
&TIM_TimeBaseStructure);

	//-------------------TIM3 PWM-----------------
----------------
	RCC_AHB1PeriphClockCmd(RCC_AHB1
Periph_GPIOC, ENABLE);	// Clocking GPIOC
(AHB1/APB1 = 42MHz)
	RCC_APB1PeriphClockCmd(RCC_APB1P
eriph_TIM3, ENABLE);	// Clocking TIM3  (APB1
= 42x2PLL=84MHz)

	//----------------Aktivasi Pin PWM-----------
----------------
	GPIO_InitStructure.GPIO_Pin	=
GPIO_Pin_6 | GPIO_Pin_7;		// Ch.1 (PC6),
Ch.2 (PC7)
	GPIO_InitStructure.GPIO_Mode	=
GPIO_Mode_AF;			// PWM is an
alternative function
	GPIO_InitStructure.GPIO_Speed =
GPIO_Speed_100MHz;		//
GPIO_HIGH_Speed
	GPIO_InitStructure.GPIO_OType =
GPIO_OType_OD;			// Push-pull
	GPIO_InitStructure.GPIO_PuPd	=
GPIO_PuPd_NOPULL;
	GPIO_Init(GPIOC, &GPIO_InitStructure);
// Initializing GPIOC structure
```

```
	GPIO_PinAFConfig(GPIOC, GPIO_PinSource6, GPIO_AF_TIM3);     // Routing TIM3 output to PC6
	GPIO_PinAFConfig(GPIOC, GPIO_PinSource7, GPIO_AF_TIM3);     // Routing TIM3 output to PC7

	// Timer3 init
	TIM_TimeBaseStructure.TIM_ClockDivision = TIM_CKD_DIV1;              // => 0 = Not dividing
	TIM_TimeBaseStructure.TIM_CounterMode   = TIM_CounterMode_Up;        // Upcounting configuration
	TIM_TimeBaseStructure.TIM_Period        = PWM_TIM3_PERIODE;          // Autoreload value (ARR)
	TIM_TimeBaseStructure.TIM_Prescaler = PWM_TIM3_PRESCALE;         // Pembagi Timclock, prescale=>328, Timclock=>84MHz
	TIM_TimeBaseInit(TIM3, &TIM_TimeBaseStructure);                  // Initializing Time Base structure

	// Channel 1, Ch.1 (PC6)
	TIM_OCInitStructure.TIM_OCMode      = TIM_OCMode_PWM1;               // PWM mode 1 = Set on compare match, PWM mode 2 = Clear on compare match pasangan OCPolarity_LOW
	TIM_OCInitStructure.TIM_OutputState = TIM_OutputState_Enable;        // Enabling the Output Compare state
	TIM_OCInitStructure.TIM_OCPolarity  = PWM_TIM3_POLARITY;             // Regular polarity (low will inverse it)
	TIM_OCInitStructure.TIM_Pulse       = PWM_R;                         // Output Compare 1 reg value
	TIM_OC1Init(TIM3, &TIM_OCInitStructure);                         // Initializing Output Compare 1 structure
	TIM_OC1PreloadConfig(TIM3, TIM_OCPreload_Enable);                // Enable Ch.1 Output Compare preload

	// Channel 2, Ch.2 (PC7)
	TIM_OCInitStructure.TIM_OCMode      = TIM_OCMode_PWM1;
	 // PWM1=Inverting=awal set(1)=polarity high, PWM2 = non inverting=awal reset(0)=polarity low
	TIM_OCInitStructure.TIM_OutputState = TIM_OutputState_Enable;        // Enabling again in case the values changed
	TIM_OCInitStructure.TIM_OCPolarity  = PWM_TIM3_POLARITY;             // => TIM_OCPolarity_High
	TIM_OCInitStructure.TIM_Pulse       = PWM_L;                         // Output Compare 2 reg value
	TIM_OC2Init(TIM3, &TIM_OCInitStructure);                         // Initializing Output Compare 2 structure
	TIM_OC2PreloadConfig(TIM3, TIM_OCPreload_Enable);                // Enable Ch.2 Output Compare preload

	// Timer enable
	TIM_Cmd(TIM3, ENABLE);
// Ready, Set, Go!
}

void USART3_PutChar(char c)
{
  uint8_t ch;
  ch = c;
  USART_SendData(USART3, (uint8_t) ch);

  /* Loop hingga transmisi selesai */
  while (USART_GetFlagStatus(USART3, USART_FLAG_TC) == RESET)
  {}
}


void usart_puts(char *data)
{
  int i=0;
  int n = strlen(data);
  for(i=0;i<n;i++)
  {
    USART3_PutChar(data[i]);
  }

}

void USART_put(USART_TypeDef* USARTx, volatile char *s){

	while(*s){
		// tunggu sampai register data SR kosong
		while( !(USARTx->SR & 0x00000040) );
		USART_SendData(USARTx, *s);
		s++;
	}
}

void usart_inisialisasi()
{
  setvbuf( stdout, 0, _IONBF, 0 );

  //--------------------------
GPIO_USART_Configuration----------------------

RCC_APB1PeriphClockCmd(RCC_APB1Periph_USART3, ENABLE);

RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPIOC, ENABLE);
  /* Configure USART Tx as alternate function  */
    GPIO_InitStructure.GPIO_OType = GPIO_OType_PP;
    GPIO_InitStructure.GPIO_PuPd = GPIO_PuPd_NOPULL;
    GPIO_InitStructure.GPIO_Mode = GPIO_Mode_AF;

    GPIO_InitStructure.GPIO_Pin = GPIO_Pin_10;
    GPIO_InitStructure.GPIO_Speed = GPIO_Speed_100MHz;
    GPIO_Init(GPIOC, &GPIO_InitStructure);

  /* Connect PXx to USARTx_Tx*/
    GPIO_PinAFConfig(GPIOC, GPIO_PinSource10, GPIO_AF_USART3);
  //--------------------------USART3_Configuration----------------
    USART_InitStructure.USART_BaudRate = 9600;
    USART_InitStructure.USART_WordLength = USART_WordLength_8b;
    USART_InitStructure.USART_StopBits = USART_StopBits_1;
    USART_InitStructure.USART_Parity = USART_Parity_No;
```

```
USART_InitStructure.USART_HardwareFlowControl
= USART_HardwareFlowControl_None;
    USART_InitStructure.USART_Mode =
USART_Mode_Tx;
    USART_Init(USART3, &USART_InitStructure);

    USART_Cmd(USART3, ENABLE); // enable
USART2

    //USART1

RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPI
OA, ENABLE);

RCC_APB2PeriphClockCmd(RCC_APB2Periph_USA
RT1,ENABLE);

    GPIO_InitStructure.GPIO_Pin =  GPIO_Pin_9 |
GPIO_Pin_10;
    GPIO_InitStructure.GPIO_Speed =
GPIO_Speed_50MHz;
    GPIO_InitStructure.GPIO_Mode =
GPIO_Mode_AF;
    GPIO_InitStructure.GPIO_OType =
GPIO_OType_PP;
    GPIO_InitStructure.GPIO_PuPd =
GPIO_PuPd_UP;
    GPIO_Init(GPIOA, &GPIO_InitStructure);

    GPIO_PinAFConfig(GPIOA, GPIO_PinSource9,
GPIO_AF_USART1);
    GPIO_PinAFConfig(GPIOA, GPIO_PinSource10,
GPIO_AF_USART1);

    USART_InitStructure.USART_BaudRate =
115200;
    USART_InitStructure.USART_WordLength =
USART_WordLength_8b;
    USART_InitStructure.USART_StopBits =
USART_StopBits_1;
    USART_InitStructure.USART_Parity =
USART_Parity_No;

USART_InitStructure.USART_HardwareFlowControl
= USART_HardwareFlowControl_None;
    USART_InitStructure.USART_Mode =
USART_Mode_Tx | USART_Mode_Rx;
    USART_Init(USART1, &USART_InitStructure);

    USART_Init(USART1, &USART_InitStructure);

    USART_ITConfig(USART1, USART_IT_RXNE,
ENABLE);

    NVIC_InitStructure.NVIC_IRQChannel =
USART1_IRQn;

NVIC_InitStructure.NVIC_IRQChannelPreemptionPrio
rity = 0;
    NVIC_InitStructure.NVIC_IRQChannelSubPriority
= 0;
    NVIC_InitStructure.NVIC_IRQChannelCmd =
ENABLE;
    NVIC_Init(&NVIC_InitStructure);

    USART_Cmd(USART1,ENABLE);
}

void EXTILine0_Config(void)
{
  /* Enable clock for GPIOA */

RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPI
OA, ENABLE);
  /* Enable clock for SYSCFG */

RCC_APB2PeriphClockCmd(RCC_APB2Periph_SYS
CFG, ENABLE);

  /* Set pin as input */
  GPIO_InitStructure.GPIO_Mode = GPIO_Mode_IN;
  GPIO_InitStructure.GPIO_OType =
GPIO_OType_PP;
  GPIO_InitStructure.GPIO_Pin = GPIO_Pin_0;
  GPIO_InitStructure.GPIO_PuPd = GPIO_PuPd_UP;
  GPIO_InitStructure.GPIO_Speed =
GPIO_Speed_100MHz;
  GPIO_Init(GPIOA, &GPIO_InitStructure);

  /* Tell system that you will use PA0 for EXTI_Line0
*/

SYSCFG_EXTILineConfig(EXTI_PortSourceGPIOA,
EXTI_PinSource0);

  /* PD0 is connected to EXTI_Line0 */
  EXTI_InitStructure.EXTI_Line = EXTI_Line0;
  /* Enable interrupt */
  EXTI_InitStructure.EXTI_LineCmd = ENABLE;
  /* Interrupt mode */
  EXTI_InitStructure.EXTI_Mode =
EXTI_Mode_Interrupt;
  /* Triggers on rising and falling edge */
  EXTI_InitStructure.EXTI_Trigger =
EXTI_Trigger_Falling;
  /* Add to EXTI */
  EXTI_Init(&EXTI_InitStructure);

  /* Add IRQ vector to NVIC */
  /* PA0 is connected to EXTI_Line0, which has
EXTI0_IRQn vector */
  NVIC_InitStructure.NVIC_IRQChannel =
EXTI0_IRQn;
  /* Set priority */

NVIC_InitStructure.NVIC_IRQChannelPreemptionPrio
rity = 0x00;
  /* Set sub priority */
  NVIC_InitStructure.NVIC_IRQChannelSubPriority
= 0x00;
  /* Enable interrupt */
  NVIC_InitStructure.NVIC_IRQChannelCmd =
ENABLE;
  /* Add to NVIC */
  NVIC_Init(&NVIC_InitStructure);
}

void EXTILine1_Config(void)
{

  /* Enable clock for GPIOA */

RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPI
OA, ENABLE);
  /* Enable clock for SYSCFG */

RCC_APB2PeriphClockCmd(RCC_APB2Periph_SYS
CFG, ENABLE);

  /* Set pin as input */
  GPIO_InitStructure.GPIO_Mode = GPIO_Mode_IN;
  GPIO_InitStructure.GPIO_OType =
GPIO_OType_PP;
  GPIO_InitStructure.GPIO_Pin = GPIO_Pin_1;
```

```
    GPIO_InitStructure.GPIO_PuPd = GPIO_PuPd_UP;
    GPIO_InitStructure.GPIO_Speed =
GPIO_Speed_100MHz;
    GPIO_Init(GPIOA, &GPIO_InitStructure);

    /* Tell system that you will use PD1 for EXTI_Line1
*/

SYSCFG_EXTILineConfig(EXTI_PortSourceGPIOA,
EXTI_PinSource1);

    /* PD1 is connected to EXTI_Line1 */
    EXTI_InitStructure.EXTI_Line = EXTI_Line1;
    /* Enable interrupt */
    EXTI_InitStructure.EXTI_LineCmd = ENABLE;
    /* Interrupt mode */
    EXTI_InitStructure.EXTI_Mode =
EXTI_Mode_Interrupt;
    /* Triggers on rising and falling edge */
    EXTI_InitStructure.EXTI_Trigger =
EXTI_Trigger_Falling;
    /* Add to EXTI */
    EXTI_Init(&EXTI_InitStructure);

    /* Add IRQ vector to NVIC */
    /* PD0 is connected to EXTI_Line1, which has
EXTI1_IRQn vector */
    NVIC_InitStructure.NVIC_IRQChannel =
EXTI1_IRQn;
    /* Set priority */

NVIC_InitStructure.NVIC_IRQChannelPreemptionPrio
rity = 0x00;
    /* Set sub priority */
    NVIC_InitStructure.NVIC_IRQChannelSubPriority
= 0x01;
    /* Enable interrupt */
    NVIC_InitStructure.NVIC_IRQChannelCmd =
ENABLE;
    /* Add to NVIC */
    NVIC_Init(&NVIC_InitStructure);
}
```

LAMPIRAN 4

DATASHEET

User's Manual 2006-06-14

DYNAMIXEL **AX-12** ROBOTIS

1-2. Main Specifications

	AX-12	
Weight (g)	55	
Gear Reduction Ratio	1/254	
Input Voltage (V)	at 7V	at 10V
Final Max Holding Torque(kgf.cm)	12	16.5
Sec/60degree	0.269	0.196

Resolution	0.35°
Operating Angle	300°, Endless Turn
Voltage	7V~10V (Recommended voltage: 9.6V)
Max. Current	900mA
Operate Temperature	-5℃ ~ +85℃
Command Signal	Digital Packet
Protocol Type	Half duplex Asynchronous Serial Communication (8bit,1stop,No Parity)
Link (Physical)	TTL Level Multi Drop (daisy chain type Connector)
ID	254 ID (0~253)
Communication Speed	7343bps ~ 1 Mbps
Feedback	Position, Temperature, Load, Input Voltage, etc.
Material	Engineering Plastic

3-2. Instruction Packet

The Instruction Packet is the packet sent by the main controller to the Dynamixel units to send commands. The structure of the Instruction Packet is as the following.

Instruction Packet | 0XFF | 0XFF | ID | LENGTH | INSTRUCTION | PARAMETER1 | ... | PARAMETER N | CHECK SUM |

The meanings of each packet byte definition are as the following.

0XFF 0XFF — The two 0XFF bytes indicate the start of an incoming packet.

ID — The unique ID of a Dynamixel unit. There are 254 available ID values, ranging from 0X00 to 0XFD.

Broadcasting ID — ID 0XFE is the Broadcasting ID which indicates all of the connected Dynamixel units. Packets sent with this ID apply to all Dynamixel units on the network. Thus packets sent with a broadcasting ID will not return any status packets.

LENGTH — The length of the packet where its value is "Number of parameters (N) + 2"

INSTRUCTION — The instruction for the Dynamixel actuator to perform.

PARAMETER0...N — Used if there is additional information needed to be sent other than the instruction itself.

CHECK SUM — The computation method for the 'Check Sum' is as the following.

Check Sum = ~ (ID + Length + Instruction + Parameter1 + ... Parameter N)
If the calculated value is larger than 255, the lower byte is defined as the checksum value.
~ represents the NOT logic operation.

3-3. Status Packet(Return Packet)

The Status Packet is the response packet from the Dynamixel units to the Main Controller after receiving an instruction packet. The structure of the status packet is as the following.

| 0XFF | 0XFF | ID | LENGTH | ERROR | PARAMETER1 | PARAMETER2 | ... | PARAMETER N | CHECK SUM |

3-4. Control Table

	Address	Item	Access	Initial Value
EEPROM Area	0(0X00)	Model Number(L)	RD	12(0x0C)
	1(0X01)	Model Number(H)	RD	0(0x00)
	2(0X02)	Version of Firmware	RD	?
	3(0X03)	ID	RD,WR	1(0x01)
	4(0X04)	Baud Rate	RD,WR	1(0x01)
	5(0X05)	Return Delay Time	RD,WR	250(0xFA)
	6(0X06)	CW Angle Limit(L)	RD,WR	0(0x00)
	7(0X07)	CW Angle Limit(H)	RD,WR	0(0x00)
	8(0X08)	CCW Angle Limit(L)	RD,WR	255(0xFF)
	9(0X09)	CCW Angle Limit(H)	RD,WR	3(0x03)
	10(0x0A)	(Reserved)	-	0(0x00)
	11(0X0B)	the Highest Limit Temperature	RD,WR	85(0x55)
	12(0X0C)	the Lowest Limit Voltage	RD,WR	60(0X3C)
	13(0X0D)	the Highest Limit Voltage	RD,WR	190(0xBE)
	14(0X0E)	Max Torque(L)	RD,WR	255(0XFF)
	15(0X0F)	Max Torque(H)	RD,WR	3(0x03)
	16(0X10)	Status Return Level	RD,WR	2(0x02)
	17(0X11)	Alarm LED	RD,WR	4(0x04)
	18(0X12)	Alarm Shutdown	RD,WR	4(0x04)
	19(0X13)	(Reserved)	RD,WR	0(0x00)
	20(0X14)	Down Calibration(L)	RD	?
	21(0X15)	Down Calibration(H)	RD	?
	22(0X16)	Up Calibration(L)	RD	?
	23(0X17)	Up Calibration(H)	RD	?
RAM Area	24(0X18)	Torque Enable	RD,WR	0(0x00)
	25(0X19)	LED	RD,WR	0(0x00)
	26(0X1A)	CW Compliance Margin	RD,WR	0(0x00)
	27(0X1B)	CCW Compliance Margin	RD,WR	0(0x00)
	28(0X1C)	CW Compliance Slope	RD,WR	32(0x20)
	29(0X1D)	CCW Compliance Slope	RD,WR	32(0x20)
	30(0X1E)	Goal Position(L)	RD,WR	[Addr36]value
	31(0X1F)	Goal Position(H)	RD,WR	[Addr37]value
	32(0X20)	Moving Speed(L)	RD,WR	0
	33(0X21)	Moving Speed(H)	RD,WR	0
	34(0X22)	Torque Limit(L)	RD,WR	[Addr14] value
	35(0X23)	Torque Limit(H)	RD,WR	[Addr15] value
	36(0X24)	Present Position(L)	RD	?
	37(0X25)	Present Position(H)	RD	?
	38(0X26)	Present Speed(L)	RD	?
	39(0X27)	Present Speed(H)	RD	?
	40(0X28)	Present Load(L)	RD	?
	41(0X29)	Present Load(H)	RD	?
	42(0X2A)	Present Voltage	RD	?
	43(0X2B)	Present Temperature	RD	?
	44(0X2C)	Registered Instruction	RD,WR	0(0x00)
	45(0X2D)	(Reserved)	-	0(0x00)
	46(0x2E)	Moving	RD	0(0x00)
	47(0x2F)	Lock	RD,WR	0(0x00)
	48(0x30)	Punch(L)	RD,WR	32(0x20)
	49(0x31)	Punch(H)	RD,WR	0(0x00)

UM1472
User Manual
STM32F4DISCOVERY
STM32F4 high-performance discovery board

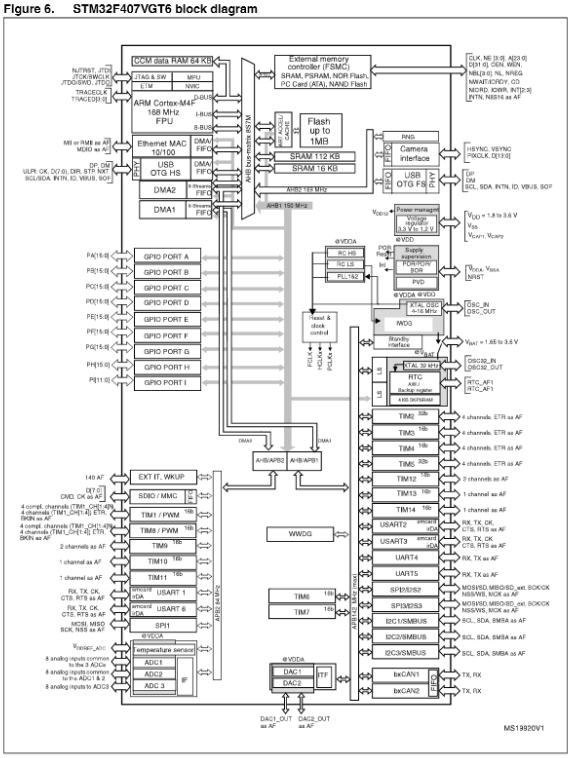
Introduction

The STM32F4DISCOVERY helps you to discover the STM32F4 high-performance features and to develop your applications. It is based on an STM32F407VGT6 and includes an ST-LINK/V2 embedded debug tool interface, ST MEMS digital accelerometer, ST MEMS digital microphone, audio DAC with integrated class D speaker driver, LEDs, pushbuttons and a USB OTG micro-AB connector.

Figure 1. STM32F4DISCOVERY

January 2012 Doc ID 022256 Rev 2 1/38

Figure 6. STM32F407VGT6 block diagram

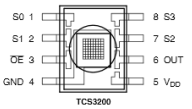


TAOS TEXAS ADVANCED OPTOELECTRONIC SOLUTIONS®

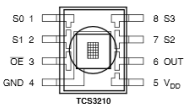
TCS3200, TCS3210
PROGRAMMABLE
COLOR LIGHT-TO-FREQUENCY CONVERTER
TAOS099 – JULY 2009

- **High-Resolution Conversion of Light Intensity to Frequency**
- **Programmable Color and Full-Scale Output Frequency**
- **Communicates Directly With a Microcontroller**
- **Single-Supply Operation (2.7 V to 5.5 V)**
- **Power Down Feature**
- **Nonlinearity Error Typically 0.2% at 50 kHz**
- **Stable 200 ppm/°C Temperature Coefficient**
- **Low-Profile Lead (Pb) Free and RoHS Compliant Surface-Mount Package**

PACKAGE D
8-LEAD SOIC
(TOP VIEW)



TCS3200



TCS3210

Description

The TCS3200 and TCS3210 programmable color light-to-frequency converters that combine configurable silicon photodiodes and a current-to-frequency converter on a single monolithic CMOS integrated circuit. The output is a square wave (50% duty cycle) with frequency directly proportional to light intensity (irradiance).

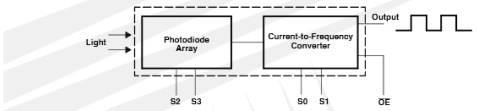
The full-scale output frequency can be scaled by one of three preset values via two control input pins. Digital inputs and digital output allow direct interface to a microcontroller or other logic circuitry. Output enable (OE) places the output in the high-impedance state for multiple-unit sharing of a microcontroller input line.

In the TCS3200, the light-to-frequency converter reads an 8 x 8 array of photodiodes. Sixteen photodiodes have blue filters, 16 photodiodes have green filters, 16 photodiodes have red filters, and 16 photodiodes are clear with no filters.

In the TCS3210, the light-to-frequency converter reads a 4 x 6 array of photodiodes. Six photodiodes have blue filters, 6 photodiodes have green filters, 6 photodiodes have red filters, and 6 photodiodes are clear with no filters.

The four types (colors) of photodiodes are interdigitated to minimize the effect of non-uniformity of incident irradiance. All photodiodes of the same color are connected in parallel. Pins S2 and S3 are used to select which group of photodiodes (red, green, blue, clear) are active. Photodiodes are 110 µm x 110 µm in size and are on 134-µm centers.

Functional Block Diagram



Texas Advanced Optoelectronic Solutions Inc.
1001 Klein Road • Suite 300 • Plano, TX 75074 • (972) 673-0759
www.taosinc.com

Terminal Functions

TERMINAL NAME	TERMINAL NO.	I/O	DESCRIPTION
GND	4		Power supply ground. All voltages are referenced to GND.
OE	3	I	Enable for f_o (active low).
OUT	6	O	Output frequency (f_o).
S0, S1	1, 2	I	Output frequency scaling selection inputs.
S2, S3	7, 8	I	Photodiode type selection inputs.
V_{DD}	5		Supply voltage

Table 1. Selectable Options

S0	S1	OUTPUT FREQUENCY SCALING (f_o)
L	L	Power down
L	H	2%
H	L	20%
H	H	100%

S2	S3	PHOTODIODE TYPE
L	L	Red
L	H	Blue
H	L	Clear (no filter)
H	H	Green

Available Options

DEVICE	T_A	PACKAGE – LEADS	PACKAGE DESIGNATOR	ORDERING NUMBER
TCS3200	–40°C to 85°C	SOIC–8	D	TCS3200D
TCS3210	–40°C to 85°C	SOIC–8	D	TCS3210D

Absolute Maximum Ratings over operating free-air temperature range (unless otherwise noted)†

Supply voltage, V_{DD} (see Note 1) . . . . . 6 V
Input voltage range, all inputs, V_I . . . . . –0.3 V to V_{DD} + 0.3 V
Operating free-air temperature range, T_A (see Note 2) . . . . . –40°C to 85°C
Storage temperature range (see Note 2) . . . . . –40°C to 85°C
Solder conditions in accordance with JEDEC J–STD–020A, maximum temperature (see Note 3) . . . 260°C

† Stresses beyond those listed under "absolute maximum ratings" may cause permanent damage to the device. These are stress ratings only, and functional operation of the device at these or any other conditions beyond those indicated under "recommended operating conditions" is not implied. Exposure to absolute-maximum-rated conditions for extended periods may affect device reliability.

NOTES: 1. All voltage values are with respect to GND.
2. Long-term storage or operation above 70°C could cause package yellowing that will lower the sensitivity to wavelengths < 500nm.
3. The device may be hand soldered provided that heat is applied only to the solder pad and no contact is made between the tip of the solder iron and the device lead. The maximum time heat should be applied to the device is 5 seconds.

Recommended Operating Conditions

		MIN	NOM	MAX	UNIT
Supply voltage, V_{DD}		2.7	5	5.5	V
High-level input voltage, V_{IH}	V_{DD} = 2.7 V to 5.5 V	2		V_{DD}	V
Low-level input voltage, V_{IL}	V_{DD} = 2.7 V to 5.5 V	0		0.8	V
Operating free-air temperature range, T_A		–40		70	°C

Electrical Characteristics at T_A = 25°C, V_{DD} = 5 V (unless otherwise noted)

PARAMETER		TEST CONDITIONS	MIN	TYP	MAX	UNIT
V_{OH}	High-level output voltage	I_{OH} = –2 mA	4	4.5		V
V_{OL}	Low-level output voltage	I_{OL} = 2 mA		0.25	0.40	V
I_{IH}	High-level input current				5	µA
I_{IL}	Low-level input current				5	µA
I_{DD}	Supply current	Power-on mode		1.4	2	mA
		Power-down mode			0.1	µA
	Full-scale frequency (See Note 4)	S0 = H, S1 = H	500	600		kHz
		S0 = H, S1 = L	100	120		kHz
		S0 = L, S1 = H	10	12		kHz
	Temperature coefficient of responsivity	λ = 700 nm, –25°C ≤ T_A ≤ 70°C		±200		ppm/°C
K_{SVS}	Supply voltage sensitivity	V_{DD} = 5 V ±10%		±0.5		%/V

NOTE 4: Full-scale frequency is the maximum operating frequency of the device without saturation.

FAIRCHILD SEMICONDUCTOR®

October 2005

BSS138

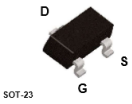
N-Channel Logic Level Enhancement Mode Field Effect Transistor

General Description

These N-Channel enhancement mode field effect transistors are produced using Fairchild's proprietary, high cell density, DMOS technology. These products have been designed to minimize on-state resistance while provide rugged, reliable, and fast switching performance.These products are particularly suited for low voltage, low current applications such as small servo motor control, power MOSFET gate drivers, and other switching applications.

Features

- 0.22 A, 50 V. $R_{DS(ON)}$ = 3.5Ω @ V_{GS} = 10 V
  $R_{DS(ON)}$ = 6.0Ω @ V_{GS} = 4.5 V
- High density cell design for extremely low $R_{DS(ON)}$
- Rugged and Reliable
- Compact industry standard SOT-23 surface mount package



SOT-23

Absolute Maximum Ratings T_A=25°C unless otherwise noted

Symbol	Parameter		Ratings	Units
V_{DSS}	Drain-Source Voltage		50	V
V_{GSS}	Gate-Source Voltage		±20	V
I_D	Drain Current – Continuous	(Note 1)	0.22	A
	– Pulsed		0.88	
P_D	Maximum Power Dissipation	(Note 1)	0.36	W
	Derate Above 25°C		2.8	mW/°C
T_J, T_{STG}	Operating and Storage Junction Temperature Range		–55 to +150	°C
T_L	Maximum Lead Temperature for Soldering Purposes, 1/16" from Case for 10 Seconds		300	°C

Thermal Characteristics

$R_{\theta JA}$	Thermal Resistance, Junction-to-Ambient	(Note 1)	350	°C/W

Package Marking and Ordering Information

Device Marking	Device	Reel Size	Tape width	Quantity
SS	BSS138	7"	8mm	3000 units

Electrical Characteristics

T_A = 25°C unless otherwise noted

Symbol	Parameter	Test Conditions	Min	Typ	Max	Units
Off Characteristics						
BV_{DSS}	Drain–Source Breakdown Voltage	V_{GS} = 0 V, I_D = 250 µA	50			V
$\frac{\Delta BV_{DSS}}{\Delta T_J}$	Breakdown Voltage Temperature Coefficient	I_D = 250 µA, Referenced to 25°C		72		mV/°C
I_{DSS}	Zero Gate Voltage Drain Current	V_{DS} = 50 V, V_{GS} = 0 V			0.5	µA
		V_{DS} = 50 V, V_{GS} = 0 V, T_J = 125°C			5	µA
		V_{DS} = 30 V, V_{GS} = 0 V			100	nA
I_{GSS}	Gate–Body Leakage.	V_{GS} = ±20 V, V_{DS} = 0 V			±100	nA
On Characteristics (Note 2)						
$V_{GS(th)}$	Gate Threshold Voltage	V_{DS} = V_{GS}, I_D = 1 mA	0.8	1.3	1.5	V
$\frac{\Delta V_{GS(th)}}{\Delta T_J}$	Gate Threshold Voltage Temperature Coefficient	I_D = 1 mA, Referenced to 25°C		–2		mV/°C
$R_{DS(on)}$	Static Drain–Source On–Resistance	V_{GS} = 10 V, I_D = 0.22 A		0.7	3.5	Ω
		V_{GS} = 4.5 V, I_D = 0.22 A		1.0	6.0	
		V_{GS} = 10 V, I_D = 0.22 A, T_J = 125°C		1.1	5.8	
$I_{D(on)}$	On–State Drain Current	V_{GS} = 10 V, V_{DS} ≥ 5 V	0.2			A
g_{FS}	Forward Transconductance	V_{DS} = 10V, I_D = 0.22 A	0.12	0.5		S
Dynamic Characteristics						
C_{iss}	Input Capacitance	V_{DS} = 25 V, V_{GS} = 0 V, f = 1.0 MHz		27		pF
C_{oss}	Output Capacitance			13		pF
C_{rss}	Reverse Transfer Capacitance			6		pF
R_G	Gate Resistance	V_{GS} = 15 mV, f = 1.0 MHz		9		Ω
Switching Characteristics (Note 2)						
$t_{d(on)}$	Turn–On Delay Time	V_{DD} = 30 V, I_D = 0.29 A, V_{GS} = 10 V, R_{GEN} = 6 Ω		2.5	5	ns
t_r	Turn–On Rise Time			9	18	ns
$t_{d(off)}$	Turn–Off Delay Time			20	36	ns
t_f	Turn–Off Fall Time			7	14	ns
Q_g	Total Gate Charge	V_{DS} = 25 V, I_D = 0.22 A, V_{GS} = 10 V		1.7	2.4	nC
Q_{gs}	Gate–Source Charge			0.1		nC
Q_{gd}	Gate–Drain Charge			0.4		nC
Drain–Source Diode Characteristics and Maximum Ratings						
I_S	Maximum Continuous Drain–Source Diode Forward Current				0.22	A
V_{SD}	Drain–Source Diode Forward Voltage	V_{GS} = 0 V, I_S = 0.44 A (Note 2)		0.8	1.4	V

Notes:

1. $R_{\theta JA}$ is the sum of the junction-to-case and case-to-ambient thermal resistance where the case thermal reference is defined as the solder mounting surface of the drain pins. $R_{\theta JC}$ is guaranteed by design while $R_{\theta CA}$ is determined by the user's board design.

a) 350°C/W when mounted on a minimum pad.

Scale 1 : 1 on letter size paper

2. Pulse Test: Pulse Width ≤ 300 µs, Duty Cycle ≤ 2.0%

BSS138 Rev. C(W)