UNIVERSITAS DIPONEGORO

PERANCANGAN MCU PADA PENGATURAN VOLUME CAIRAN MENGGUNAKAN *FUZZY LOGIC CONTROLLER* METODE SUGENO

TUGAS AKHIR

Diajukan sebagai salah satu syarat untuk memperoleh gelar Sarjana Teknik

YOHANES JUAN KURNIADI
21060118140124

FAKULTAS TEKNIK
DEPARTEMEN TEKNIK ELEKTRO
PROGRAM STUDI SARJANA
SEMARANG
SEPTEMBER 2022

UNIVERSITAS DIPONEGORO

PERANCANGAN MCU PADA PENGATURAN VOLUME CAIRAN MENGGUNAKAN *FUZZY LOGIC CONTROLLER* METODE SUGENO

TUGAS AKHIR

Diajukan sebagai salah satu syarat untuk memperoleh gelar Sarjana Teknik

YOHANES JUAN KURNIADI

21060118140124

FAKULTAS TEKNIK

DEPARTEMEN TEKNIK ELEKTRO

PROGRAM STUDI SARJANA

SEMARANG

SEPTEMBER 2022

HALAMAN PERNYATAAN ORISINALITAS

Tugas Akhir ini adalah karya saya sendiri,
dan semua sumber baik yang dikutip maupun yang dirujuk
telah saya nyatakan dengan benar

NAMA	:	YOHANES JUAN KURNIADI
NIM	:	21060118140124
Tanda Tangan	:	
Tanggal	:	9 September 2022

HALAMAN PENGESAHAN

Tugas Akhir ini diajukan oleh :

NAMA	: YOHANES JUAN KURNIADI
NIM	: 21060118140124
Departemen/Program Studi	: TEKNIK ELEKTRO / SARJANA (S1)
Judul Skripsi	: Perancangan MCU Pada Pengaturan *Volume* Cairan Menggunakan *Fuzzy Logic Controller* Metode Sugeno

Telah berhasil dipertahankan di hadapan Tim Penguji dan diterima sebagai bagian persyaratan yang diperlukan untuk memperoleh gelar Sarjana Teknik pada Program Studi Sarajan , Departeman Teknik Elektro, Fakultas Teknik , Universitas Diponegoro.

TIM PENGUJI

Pembimbing	: Ajub Ajulian Zahra, S.T., M.T.	( )
Pembimbing	: Ir. Sumardi, S.T., M.T., IPM.	( )
Penguji I	:	
Penguji II	:	
Penguji III	:	

Semarang, 9 September 2022
Ketua Departemen Teknik Elektro,

Aghus Sofwan, S.T., M.T.,Ph.D
NIP. 197206302000121001

HALAMAN PERNYATAAN PERSETUJUAN PUBLIKASI TUGAS AKHIR UNTUK KEPENTINGAN AKADEMIS

Sebagai sivitas akademika Universitas Diponegoro, saya yang bertanda tangan di bawah ini :

Nama	: Yohanes Juan Kurniadi
NIM	: 21060118140124
Program Studi	: SARJANA (S1)
Departemen	: TEKNIK ELEKTRO
Fakultas	: TEKNIK
Jenis Karya	: TUGAS AKHIR

Demi pengembangan ilmu pengetahuan, menyetujui untuk memberikan kepada Universitas Diponegoro **Hak Bebas Royalti Noneksklusif** (*Non-exclusive Royalty Free Right)* atas karya ilmiah saya yang berjudul : PERANCANGAN MCU PADA PENGATURAN *VOLUME* CAIRAN MENGGUNAKAN *FUZZY LOGIC CONTROLLER* METODE SUGENO. Beserta perangkat yang ada (jika diperlukan). Dengan Hak Bebas Royalti/Noneksklusif Universitas Diponegoro berhak menyimpan, mengalih media/formatkan, mengelola dalam bentuk pangkalan data (*database*), merawat dan mempublikasikan tugas akhir saya selama tetap mencantumkan nama saya sebagai penulis/pencipta dan sebagai pemilik Hak Cipta.

Demikian pernyataan ini saya buat dengan sebenarnya.

Dibuat di : Semarang
Pada Tanggal : 9 September 2022

Yang menyatakan,

(Yohanes Juan Kurniadi)
NIM. 21060118140124

ABSTRAK

Dalam upaya untuk membuat pembangunan berkelanjutan di tengah pandemi COVID-19 dibuatlah sebuah Alat & Dispenser Desinfektan dan *Hand Sanitizer* otomatis dengan menanamkan teknologi *touchless* menggunakan *Fuzzy Logic Controller* metode Sugeno sebagai pengampil keputusan. Pada penilitian ini bertujuan untuk mendukung kebiasaan tanpa setuhan. Alat Tugas Akhir ini menggunakan pompa air DC sebagai aktuator yang mengalirkan cairan – cairan dari titik satu ke titik lainnya yang sudah disesuaikan oleh peneliti. Sensor yang digunakan untuk menimbang tangki penampung adalah sensor berat (*loadcell*). Parameter yang akan diambil adalah parameter batas minimum dan maksimum untuk sensor *loadcell*, kemudian hasil *output* dari kontroler *Fuzzy Logic*. Diperoleh hasil penelitian perhitungan batas minimum untuk *loadcell* pada menit ke – 2 diperolah *error* sebesar 14,30% untuk sanitasi dan menit ke – 2 diperoleh *error* sebesar 4,43% untuk batas disinfektan. Sedangkan batas maksimum, pada menit ke – 2 diperoleh *error* sebesar 11,68% untuk sanitasi dan menit ke – 10 diperoleh *error* sebesar 11,64% untuk disinfektan. Begitupula untuk hasil perhitungan *Fuzzy Logic* yang sudah didapat untuk sanitasi nilai *output* sebesar 10,20% untuk Alkohol Sanitasi, 10,78% untuk Glycerin, dan 12,62% untuk H_2O_2. Begitupula untuk hasil nilai *output* disinfektan sebesar 12,56% untuk Alkohol Disinfektan, 15,58% untuk Bayclin, dan 10,61% untuk Wipol.

Kata kunci: *Fuzzy Logic Controller*, *loadcell*, dan pPompaair DC

ABSTRACT

To create sustainable development in the midst amid pandemic, an automatic Disinfectant and Hand Sanitizer Tool & Dispenser was created by embedding touchless technology using the Sugeno Fuzzy Logic Controller method as a decision maker. This research aims to support the habit of no god. This final project tool uses a DC water pump as an actuator that flows liquids from one point to another that has been adjusted by the researcher. The sensor used to weigh the holding tank is a weight sensor (loadcell). The parameters to be taken are the minimum and maximum limit parameters for the loadcell sensor, then the output results from the Fuzzy Logic controller. The results of the research on the calculation of the minimum limit for loadcell in the 2nd minute obtained an error of 14.30% for sanitation and the 2nd minute an error in of 4.43% was obtained for the disinfectant limit. While the maximum limit, in the 2nd minute an error of 11.68% is obtained for sanitation ain nd the 10th minute ,an error of 11.64% is obtained for the disinfectant. Likewise for the results of Fuzzy Logic calculations that have been obtained for sanitation, the output value is 10.20% for Sanitary Alcohol, 10.78% for Glycerin, and 12.62% for H2O2. Like for the results of the disinfectant output value of 12.56% for Alcohol Disinfectant, 15.58% for Bayclin, and 10.61% for Wipol.

***Keywords**: Fuzzy Logic Controller, loadcell, and DC water pump*

KATA PENGANTAR

Puji syukur penulis panjatkan kehadirat Allah Subhanahu wa Ta'ala atas rahmat, karunia, taufik, dan hidayah-Nya, sehingga penulis dapat menyelesaikan tugas akhir dan penyusunan laporan ini. Tugas Akhir berjudul "**PERANCANGAN MCU PADA PENGATURAN *VOLUME* CAIRAN MENGGUNAKAN *FUZZY LOGIC CONTROLLER* METODE SUGENO**" ini diajukan sebagai syarat akhir untuk menyelesaikan program Sarjana di Departemen Teknik Elektro, Fakultas Teknik, Universitas Diponegoro, Semarang. Penyusunan dan penyelesaian Tugas Akhir ini tidak lepas dari bantuan dan dukungan dari semua pihak, baik secara langsung maupun tidak langsung. Oleh karena itu, pada kesempatan kali ini penulis mengucapkan terima kasih kepada:

1. Bapak Agus Sofwan, S.T., M.T., Ph.D selaku Ketua Departemen Teknik Elektro Fakultas Teknik Universitas Diponegoro Semarang.
2. Bapak Dr. Munawar Agus Riyadi, S.T., M.T., selaku Ketua Program Studi S1 Teknik Elektro Universitas Diponegoro Semarang.
3. Bapak Yuli Christiyono, S.T., M.T., selaku Sekretaris Program Studi S1 Teknik Elektro Universitas Diponegoro.
4. Ibu Ajub Ajulian Zahra, S.T., M.T., selaku Dosen Pembimbing I yang selalu memberikan bimbingan, arahan, dan motivasi dalam menyelesaikan tugas akhir ini.
5. Bapak Sumardi, S.T., M.T., selaku Dosen Pembimbing II yang selalu memberikan motivasi, bimbingan, serta arahan yang membangun kepada penulis.
6. Bapak Sumardi, S.T., M.T., selaku dosen wali penulis yang selalu memberikan semangat dan mendengarkan dengan baik permasalahan penulis selama menempuh studi di Departemen Teknik Elektro Universitas Diponegoro.
7. Kedua orang tua penulis dan keluarga besar penulis yang selalu memberikan kasih sayang, doa, motivasi dan semangat kepada penulis.

8. Michelle Angelina Archan yang banyak memberikan dukungan, doa, serta penuh perhatiannya kepada penulis dalam menyelesaikan Tugas Akhir ini.
9. Sahabat kelompok tugas akhir Surya Eka Pratama dan Martinus De Parres Putu Marino yang banyak membantu penulis dalam menyelesaikan Tugas Akhir ini.
10. Keluarga Sirius de Elera S1 Teknik Elektro Universitas Diponegoro Angkatan 2018 khususnya konsentrasi Teknologi Informasi, terimakasih atas semuanya selama penulis menimba ilmu di S1 Teknik Elektro Universitas Diponegoro.
11. Untuk kontrakan Keluarga Besar Ayah Hema, banyak terimakasih atas semua bantuan dan kerja kerasnya bersama – sama banyak melewati rintangan serta bahu membahu sampai Tugas Akhir ini selesai.
12. Semua pihak yang tidak dapat penulis sebutkan satu persatu yang telah membantu dari awal hingga akhir.

Penulis menyadari bahwa dalam penulisan laporan Tugas Akhir ini tentunya ada kekurangan oleh karena itu kritik dan saran yang dapat membangun diperlukan demi kebaikan dan kesempurnaan penyusunan di masa yang akan datang. Semoga Tugas Akhir ini dapat memberikan manfaat dan menambahkan pengetahuan bagi kita semua.

Semarang, 9 September 2022

Penulis

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I
PENDAHULUAN

1.1 Latar Belakang

Perkembangan Teknologi kini berkembang pesat, dimanapun kita pergi, kita akan menemui aktivitas yang dahulu dilakukan secara tradisional maupun konvensional kini terpengaruh dengan teknologi dan informasi. Alat-alat baru yang diciptakan sangat mempermudah manusia dalam melakukan aktivitas sehari-hari. Alat-alat ini membuat manusia sangat dekat dengan Teknologi dan Informasi, kedekatan antara manusia dan teknologi menuntut banyak diciptakannya alat-alat yang fleksibel, ergonomis, dan dinamis. Dalam hal ini baterai menjadi komponen yang esensial karena dapat menyimpan energi listrik untuk menghidupkan Teknologi yang diperlukan manusia setiap waktu.

Menurut penelitian pada tahun 2018, Sartika Lina, dkk[1] telah melakukan penerapan metode *Fuzzy Inference System* (FIS), dapat digunakan untuk menentukan jumlah dan kapan waktu pembelian produk untuk persediaan. Logika *fuzzy* dianggap mampu untuk memetakan suatu input kedalam suatu output tanpa mengabaikan faktor-faktor yang ada. Logika *fuzzy* diyakini sangat fleksibel dan memiliki toleransi terhadap data-data yang ada. Berdasarkan logika *fuzzy*, akan dihasilkan suatu model dari suatu sistem yang mampu memperkirakan suhu ketinggian air pada tabung reaksi yang berupa *volume* dan berat. Ada keuntungan dari *fuzzy logic* ini untuk lebih presisi. Selain pada tahun 2018, melakukan penelitian Penerapan *Fuzzy Logic* Sugeno dan Mamdani untuk memutuskan prakiraan cuaca. Mereka mengambil tiga pengujian terhadap keanggotaan suhu, keanggotaan tekanan udara, dan keanggotaan kelembaban relatif dari segi fungsi keanggotaan serta derajat keanggotaan.

Prototipe alat dispenser otomatis menggunakan *Fuzzy Logic Controller* yang dimana lebih dikhususkan *Fuzzy* Sugeno, dimana ada berbagai *input* serta *output* dan parameter – parameter yang akan dibawa pada alat ini. *Input* yang berupa bahan – bahan alami seperti *Bayclin, Wipol, Alkohol, H_2O_2, dan Glycerin* serta bahan lainnya yang mudah didapatkan, yang kemudian diambil parameter berupa *volume* berat dari setiap tabung menggunakan sensor *load cell* yang sesuai dengan *volume* masing – masing tabung. Dari semua *input* dan parameter yang diambil akan dihasilkan *output* berupa cairan desinfektan maupun sanitasi dari pengguna/*user*.

Penggunaan kontrol *Fuzzy Logic* yang akan sangat membantu dalam hal tingkat keakuratan bahan – bahan untuk alat pencampur dan dispenser cairan sanitasi melalui rasio dari tiap – tiap bahan nantinya. Tiap – tiap bahan yang akan digunakan untuk mengetahui apakah sisa bahan cukup/tidak untuk setiap siklus nya yang akan masuk melalui tangki *buffer/mixer*. Kontrol *Fuzzy Logic* yang nantinya akan bekerja mengoptimalkan semua bahan – bahan apakah sudah sesuai dengan permintaan *user* melalui data yang tersimpan di *web server* yang akan dikontrol menggunakan Mikrokontroler ESP32 di MCU.

Dalam penelitian yang dilakukan oleh Trstenjak dan Donko (2013) [2], peneliti menggunakan logika fuzzy untuk mengevaluasi kualitas guru di HEI. Evaluasi tersebut dilakukan oleh para siswa melalui survei terstandardisasi yang pertanyaannya didefinisikan pada tingkat keadaan. Penelitian ini menunjukkan penggunaan logika fuzzy dan metode Multi-Criteria Decision Making (MCDM) berupa TOPSIS dengan tujuan melakukan modernisasi teknik evaluasi konvensional yang selama ini digunakan. Para siswa diberikan kuesioner terkait tingkat kepentingan masing-masing kriteria penilaian guru. Akhirnya, didapatkan pembobotan kriteria untuk masing-masing kriteria. Pendekatan fuzzy TOPSIS digunakan untuk menentukan peringkat kualitas para guru.

Penelitian yang dilakukan oleh Firman B. (2016)[3] juga menggunakan logika fuzzy. Dalam studi yang dilakukan, peneliti mengukur tingkat kepuasan pekerjaan (job satisfaction) menggunakan sekumpulan himpunan fuzzy. Pengukuran tingkat kepuasaan pekerjaan dilaksanakan menggunakan kumpulan

variabel dari beberapa kriteria yang didefinisikan oleh para pakar. Pada dasarnya, pengukuran berdasarkan pada evaluasi linguistik dari variabel yang terlibat. Teori himpunan fuzzy dikombinasikan dengan analisis conjoint, kemudian digunakan untuk mengukur kepuasaan pekerjaan dari para karyawan hotel. Tingkat kepuasaan pekerjaan dari para karyawan dinilai melalui kuesioner yang memiliki pertanyaan dengan skala likert bergerak dari poin 1-5 (very dissatisfied, dissatisfied, neutral, satisfied, dan very satisfied). Pendapat dari para responden dan pakar digunakan untuk mengukur kepuasaan pekerjaan karyawan hotel. Integrasi logika fuzzy dan conjoint digunakan untuk mengevaluasi kepuasaan pekerjaan dari para karyawan hotel. Berdasarkan eksperimen, didapatkan hasil bahwa dari analisis conjoint fuzzy dapat digunakan sebagai metode alternatif untuk menganalisis kepuasan pekerjaan.

Lain halnya dengan penelitian yang dilakukan oleh Maa et al. (2016) yang membahas mengenai pengaplikasian logika *fuzzy* di dalam menghadapi permasalahan terkait penilaian kepuasan pekerjaan[4]. Metode statistik dalam melakukan penilaian kepuasan dinilai tidak dapat menghitung secara nyata terkait permasalahan kepuasan seperti aktivitas, inpendensi, variasi, status sosial, dan hubungan pengawasan-sumber daya manusia. Penilaian yang ada dikatakan hanya menilai berdasarkan persepsi dan bukan berdasarkan informasi numerik. Informasi yang berupa persepsi tersebut dapat diproses menggunakan logika *fuzzy* agar lebih memadai. Dalam penelitian ini diusulkan aturan *if-then* berdasarkan sistem pakar untuk mendeskripsikan hubungan antara faktor-faktor pekerjaan dan kepuasaan pekerjaan secara keseluruhan. Tingkat kepuasaan direpresentasikan ke dalam bahasa linguistik berikut : *very satisfied, satisfied, quite satisfied, less satisfied,* dan *unsatisfied.* Selanjutnya, aturan *if-then* diimplementasikan ke dalam sistem pakar ESPLAN. Selain itu, dilakukan pula pengetesan untuk menghitung kepuasan pekerjaan dengan data riil. Hasil evaluasi kepuasaan pekerjaan menunjukkan validitas dan efisiensi dengan metode yang disarankan.

Berawal dari beberapa penelitian terdahulu yang ada, dapat disimpulkan bahwa penulis menggunakan metode Sugeno untuk alat Tugas Akhir ini dikarenakan metode Sugeno memiliki hasil keluaran berbentuk *crisp* yang akan lebih pasti untuk menentukan jumlah – jumlahnya sesuai dengan yang sudah ditentukan dalam *membership function* penulis. Pada Tugas Akhir ini dilakukan

"**Perancangan Mcu Pada Pengaturan *Volume* Cairan Menggunakan *Fuzzy Logic Controller* Metode Sugeno**", yang akan dibahas dalam bab – bab berikutnya.

1.2 Tujuan

Tujuan Tugas Akhir ini adalah :

1. Merancang sistem kontrol *volume* cairan pada dispenser otomatis menggunakan metode Sugeno.
2. Mengukur serta membaca *error* yang dihasilkan pada sensor *Loadcell*

1.3 Batasan Masalah

Dalam penyusunan penelitian ini, telah ditentukan batasan – batasan masalah sebagai berikut :

1. Data parameter *output* dari *Fuzzy Logic* hanya mengambil keputusan yang sudah disesuaikan dengan *input.*
2. Control *Fuzzy Logic* fungsinya untuk mencapai suatu *set point* yang dituju, bukan sebagai akurasi hasil *output.*
3. Tidak membahas SoC baterai beserta aplikasi.
4. Batasan minimum untuk *trigger* sensor *load cell* adalah sebesar ≤ 200 gram, sedangkan untuk batasan maksimum *trigger* sensor *load cell* adalah sebesar ≤ 1250 gram untuk kedua tabung *buffer* menggunakan versi penulis.
5. Masukan dari perancangan kontroler alat dispenser otomatis adalah *error* dari presentase tabung bahan – bahan untuk pembuatan cairan sanitasi dan disinfektan.
6. Tidak membahas model sistematis dari sistem.
7. *Input Fuzzy* berupa *error* bahan – bahan pembuat sanitasi dan disinfektan dari *range 0* hingga 100 untuk mengendalikan *output Fuzzy* yang berupa presentase volume (%).

1.4 Sistematika Penulisan

Sistematika penulisan dalam laporan penelitian ini adalah sebagai berikut:

BAB I PENDAHULUAN

Bab ini berisi latar belakang, tujuan, batasan masalah, dan sistematika penulisan.

BAB II KAJIAN PUSTAKA

Bab ini membahas mengenai dasar teori yang digunakan dalam pembuatan Tugas Akhir ini antara lain komponen – komponen yang dibutuhkan dalam pembuatan perancangan alat dispenser otomatis.

BAB III PERANCANGAN SISTEM

Bab ini berisi tentang perancangan *Fuzzy Logic Controler* menggunakan metode Sugeno sebagai monitoring hasil *output* mesin pencampur dan dispenser cairan sanitasi terkontrol melalui aplikasi *android*.

BAB IV PENGUJIAN DAN ANALISIS

Pada bab ini dilakukan pembahasan dan analisis tentang hasil pengujian komunikasi.

BAB V PENUTUP

Bab ini berisi kesimpulan dan saran terhadap hasil pengujian dan analisis yang telah dibuat.

BAB II
KAJIAN PUSTAKA

2.1 Pengaplikasian Metode *Fuzzy Logic*

Telah banyak penelitian baik yang objeknya berupa pengukuran kinerja maupun penelitian yang menggunakan logika *fuzzy* sebagai pendekatan dalam pengambilan keputusan suatu manajemen. Berbagai metode dilakukan untuk membantu manajemen sumber daya manusia dalam memberikan penilaian terhadap pegawai. Penggunaan logika *fuzzy* dianggap mampu mengurangi tingkat inkonsistensi yang muncul dalam penilaian kinerja.

Menurut hasil penelitian dengan judul "*A novel cost function based on decomposingleast-square support vector machine forTakagi–Sugeno fuzzy system identification*" [5], studi ini mengembangkan dan menggambarkan fungsi biaya baru yang merupakan kombinasi dari istilah-istilah ini yang terdiri dari penguraian mesin vektor pendukung kuadrat terkecil (LS-SVM) dan istilah kesalahan untuk identifikasi sistem fuzzy Takagi–Sugeno (T–S) hanya berdasarkan pengukuran data tanpa pengetahuan sebelumnya. Metode yang diusulkan menggabungkan keunggulan teori sistem fuzzy dan beberapa ide dari LS-SVM.

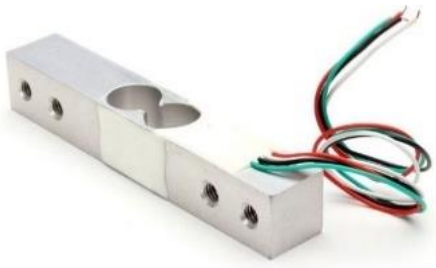
Penelitian yang dilakukan oleh, Raad Z. Homad, Khairul Salleh Mohamed Sahari, et al "*Gradient auto-tuned Takagi–Sugeno Fuzzy Forward control of a HVAC system using predicted mean vote index*", Pengendali sistem HVAC diharapkan dapat memanipulasi karakteristik nonlinier yang melekat dari sistem skala besar ini yang juga memiliki waktu jeda murni, inersia termal yang besar, faktor gangguan yang tidak pasti dan kendala[6]. Selain itu, kenyamanan termal dalam ruangan dipengaruhi oleh suhu dan kelembaban, yang adalah properti yang digabungkan. Untuk mengontrol karakteristik yang digabungkan ini dan mengatasi nonlinier secara efektif, ini makalah mengusulkan strategi kontrol Takagi–Sugeno Fuzzy Forward (TSFF) yang disetel secara online. Model TSnya adalah pertama

kali dilatih secara offline menggunakan algoritma Gauss–Newton Method for Nonlinear Regression (GNMNR) dengan data dikumpulkan dari peralatan gedung dan sistem HVAC. Model kemudian disetel secara online menggunakan algoritma gradien untuk meningkatkan stabilitas sistem secara keseluruhan dan menolak gangguan dan ketidakpastian efek.

Dalam Tugas Akhir ini, peneliti merancang sistem untuk pengaturan *volume* cairan menggunakan *Fuzzy Logic Control* metode Sugeno berdasarkan dari referensi – referensi yang sudah diambil. Kemudian, dalam bab ini akan dibahas mulai dari hardware yang digunakan sampai software yang digunakan.

2.2 Sensor *Load cell*

Sensor *load cell* merupakan sensor yang dirancang untuk mendeteksi tekanan atau berat sebuah beban, sensor *load cell* umumnya digunakan sebagai komponen utama pada sistem timbangan digital dan dapat diaplikasikan pada jembatan timbangan yang berfungsi untuk menimbang berat dari truk pengangkut bahan baku, pengukuran yang dilakukan oleh *load cell* menggunakan prinsip tekanan[7]. Gambar 2.1 menunjukkan bentuk fisik sensor *load cell* / sensor berat.



Gambar 2.1. Bentuk fisik *load cell*

Keterangan gambar :

1) Kabel merah adalah *input* tegangan sensor.
2) Kabel hitam adalah *input ground* sensor.
3) Kabel hijau adalah *output* positif sensor.
4) Kabel putih adalah *output ground* sensor.

Prinsip Kerja Sensor Berat (*load cell*), Selama proses penimbangan akan mengakibatkan reaksi terhadap elemen logam pada *load cell* yang mengakibatkan

gaya secara elastis. Gaya yang ditimbulkan oleh regangan ini dikonversikan kedalam sinyal elektrik oleh strain gauge (pengukur regangan) yang terpasang pada *load cell*. Berikut merupakan spesifikasi dari sensor *load cell* ditunjukkan pada tabel 2.1[8].

Tabel 2.1. Spesifikasi sensor *load cell*

Spesifikasi	Keterangan
Housing Material	Aluminium Alloy
Load cell Type	Strain Gauge
Cable Length	550 mm
Capacity	5 kg
Precision	0.05%
Excitation Voltage	5 VDC
Safe Overload	120% Capacity
Ultimate Overload	150% Capacity
Dimensions	55.25x12.7x12.7mm
Input Impedance	1130±10 Ohm
Output Impedance	1000±10 Ohm

2.3 Modul Penguat HX711

HX711 adalah sebuah komponen terintegrasi dari "AVIA SEMICONDUCTOR", HX711 presisi 24-bit analog to digital conventer (ADC) yang didesain untuk sensor timbangan digital dal industrial control aplikasi yang terkoneksi sensor jembatan. HX711 adalah modul timbangan, yang memiliki prinsip kerja mengkonversi perubahan yang terukur dalam perubahan resistansi dan mengkonversinya ke dalam besaran tegangan melalui rangkaian yang ada[7]. Gambar 2.2 menunjukkan modul penguat HX711 untuk *Load cell*.

Gambar 2.2. Modul penguat HX711

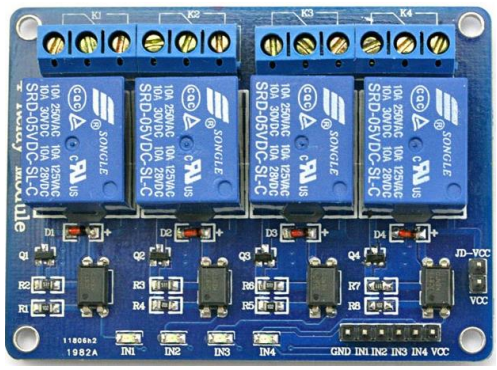
Modul melakukan komunikasi dengan komputer/mikrokontroler melalui TTL232. Struktur yang sederhana, mudah dalam penggunaan, hasil yang stabil dan reliable, memiliki sensitivitas tinggi, dan mampu mengukur perubahan dengan cepat. Memiliki 1 pin *output* Pin SCK/ sinyal clock dan memiliki 1 pin *input* DT/data. Berikut merupakan spesifikasi modul HX711 ditunjukkan pada tabel 2.2[9].

Tabel 2.2. Spesifikasi modul HX711

Nama	Fungsi	Keterangan
VS	*Power*	*Regulator Supply*: 2.7 ~ 5.5V
BASE	*Analog Output*	*Regulator control output (NC when not used)*
AVDD	*Power*	*Analog supply*: 2.6 ~ 5.5V
VFB	*Analog Input*	*Regulator control input (connect to AGND when not used*）
AGND	*Ground*	*Analog Ground*
VBG	*Analog Output*	*Reference bypass output*
INA-	*Analog Input*	*Channel A negative input*
INA+	*Analog Input*	*Channel A positive input*
INB-	*Analog Input*	*Channel B negative input*
INB+	*Analog Input*	*Channel B positive input*
PD_SCK	*Digital Input*	*Power down control (high active) and serial clock input*
DOUT	*Digital Output*	*Serial data output*
XO	*Digital I/O*	*Crystal I/O (NC when not used)*
XI	*Digital Input*	*Crystal I/O or external clock input*, 0: *use an on-chip oscillator*
RATE	*Digital Input*	*Output data rate control*, 0: 10Hz; 1: 80Hz
DVDD	*Power*	*Digital supply*: 2.6 ~ 5.5V

2.4 Relai 12V

Relai adalah perangkat yang menghubungkan kabel ke inti dengan saklar tegangan rendah. Dalam pembahasan ini, peneliti menggunakan relai 2 channel, relai jenis ini sudah menggunakan octocoupler sehingga lebih aman untuk penggunaan jangka panjang. Relai berfungsi untuk memutus dan menyambungkan listrik untuk rangkaian secara otomatis, mengendalikan tegangan tinggi menggunakan tegangan rendah, meminimalisir terjadinya penurunan tegangan, memungkinkan fungsi penundaan waktu/jeda waktu, dan melindungi komponen lain dari korsleting akibat kelebihan tegangan. Untuk penggunaan pada arduino, relayi membutuhkan 2 (dua) kabel sebagai input dan 2 (dua) kabel lagi untuk suplay tegangan[10]. Gambar 2.3 merupakan gambar bentuk fisik relai 12v.



Gambar 2.3. Bentuk fisik modul relai 12V

Spesifikasi[10]:

1. Modul ini menggunakan relai asli berkualitas tipe Normally Open (NO) dengan maximum load AC 250V/10A, DC 30V/10A
2. Memakai SMD Optocoupler isolation, yang berkinerja stabil dengan arus pemicu (trigger current) hanya sebesar 5mA
3. Tegangan sinyal pemicu sebesar 5V DC 9
4. Dapat disetting untuk mendeteksi high atau low dengan mengubah jumper
5. Dirancang dengan toleransi keamanan, bahkan jika arus pemicu putus, relay tidak akan bekerja
6. Dilengkapi lampu indikator Power (hijau) dan Status Relay (merah)

7. Mudah dipasang, menggunakan terminal untuk pemasangan kabel.
8. Ukuran: 50x41x18.5mm
9. Dilengkapi 4 lobang baut berdiameter 3.1mm berjarak 44.5mm x 35.5mm

Interface pemicu:

1. DC+: power +5V DC
2. DC-: power -5V DC
3. IN1: sinyal low atau high pada channel 1 (stel jumper)
4. IN2: sinyal low atau high pada channel 2 (stel jumper)

Interface output relay:

1. NO1: normally open channel 1 (relay tidak hidup sampai ada sinyal baru hidup)
2. COM1: common interface channel 1
3. NC1: normally close channel 1 (relay hidup sampai ada sinyal baru mati)
4. NO2: normally open channel 2 (relay tidak hidup sampai ada sinyal baru hidup)
5. COM2: common interface channel 2
6. NC2: normally close channel 2 (relay hidup sampai ada sinyal baru mati)

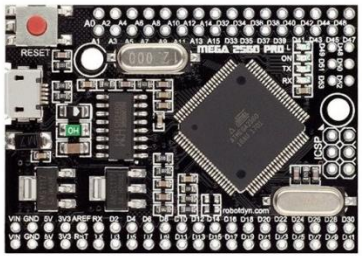
2.5 Aktuator Pompa DC

Pompa air DC berfungsi untuk memindahkan suatu cairan dari satu tempat ke tempat lainnya menggunakan sumber listrik DC. Besar daya yang digunakan pompa ini adalah 12 V dengan arus listrik sebesar 0,5 A hingga 0,7 A, kemudian ketika pompa dalam keadaan *standby*, daya yang dibutuhkan adalah 6 V dengan arus sebesar 0,18 A[11]. Gambar 2.4 merupakan pompa air DC yang digunakan pada Tugas Akhir ini.

Gambar 2.4. Pompa air DC

2.6 Mikrokontroler ATMega2560 Pro Mini

Board mikrokontroler ATMega2560 Pro Mini adalah sebuah board Arduino bersifat *open-source* yang menggunakan IC ATMega2560 jenis AVR dari perusahaan Atmel sebagai mikrokontroler pengolah data. *Board* ini biasa disebut Arduino Mega2560. Arduino Mega 2560 memiliki 54 digital pin *input/output*, 16 pin diantaranya digunakan sebagai analog *input*, 15 pin digunakan sebagai *output* PWM, 4 pin UART, *jack power*, koneksi USB, osilator kristal 16 MHz, tombol reset, dan soket *In Circuit System Programming* (ICSP)[12]. Gambar 2.5 menunjukan board mikrokontroler ATMega 2560 Pro Mini.



Gambar 2.5. Board miktrokontroler ATMega2560 Promini

Board ini memiliki *pin* digital yang dapat digunakan sebagai *input* maupun *output*. Tegangan *output* yang dihasilakn setiap *pin* adalah 5 V dengan arus maksimal 40 mA. Pada *board* mikrokontroler ATMega2560 ini juga terdapat 16 buah *input* analog (ADC) dengan setiap *input* mempunyai resolusi sebesar 10 bit. ATMega2560 memiliki 4 pin UART yang dapat digunakan sebagai komunikasi *serial* TTL, dilengkapi juga dengan TTL LED Rx dan Tx pada *board* yang akan

menyala apabila terdapat data yang dikirim atau diterima oleh *board*. Berikut merupakan spesifikasi dari *board* mikrokontroler ATMega2560 ditunjukan oleh Tabel 2.3[12].

Tabel 2.3. Spesifikasi Board Mikrokontroler ATMega2560 Pro Mini

Spesifikasi	Keterangan
Mikrokontroler	ATMega2560
Tegangan operasi	5 V
Tegangan *input* (rekomendasi)	7 - 12 volt
Tegangan *input* (limit)	6 - 20 volt
Pin Digital *Input/Output*	54 pin (15 pin untuk *output* PWM)
Arus DC per *Pin Input/Output*	20 mA
Arus DC *Pin* 3.3volt	50mA
Pin Analog *Input*	16 *pin*
Memori Flash	256 kb (8 kb untuk *bootloader*)
EEPROM	4 kb
SRAM	8 kb
Clock Speed	16 MHz
LED_BUILTIN	13
Panjang fisik	101.52 mm
Lebar fisik	53.3 mm
Berat benda	37 gr

2.7 Arduino IDE

Arduino *Integrated Development Environment* (IDE) merupakan sebuah *software* atau perangkat lunak yang digunakan oleh *editor* untuk melakukan penulisan dan penyusunan program ke mikrokontroler AVR. *Software* ini menggunakan bahasa C/C++ untuk memprogram. Interface pada Android IDE ditunjukan pada gambar 2.6, pada bagian atas terdapat *tools* seperti (dari kanan) *verify, upload, new sketch, open sketch,* dan *save sketch.* Pada bagian bawah terlihat

message box berwarna hitam berisikan status *error*, *compile*, atau status program berhasil di-*upload*.

Gambar 2.6. Tampilan pada Arduino IDE

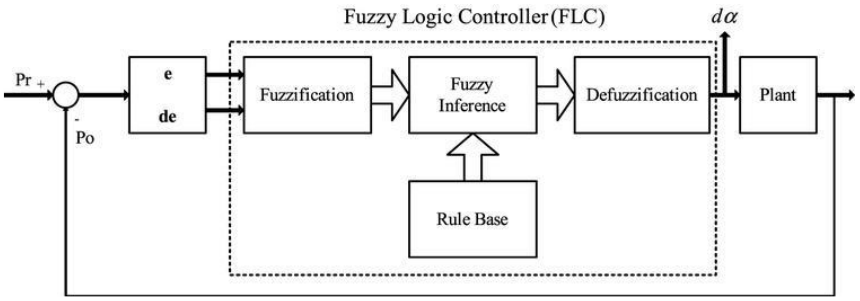
Arduino IDE memiliki 2 buah fungsi utama yaitu: program *setup* diawali dengan fungsi *void setup()* adalah inisialisasi awal yang akan diolah pertama saat program dijalankan dan hanya dijalankan sekali saja dan program *loop* yang diawali dengan fungsi *void loop()* adalah fungsi yang akan dijalankan secara terus menerus hingga kondisi mikrokontroler mati.

2.8 Kontrol Volume Menggunakan Logika *Fuzzy*

2.8.1 Fuzzy Inference System (FIS)

Pengendali *Fuzzy* atau disebut juga *Fuzzy Logic Controller (FLC)* adalah salah satu metode kontrol yang menggunakan himpunan *fuzzy (fuzzy set)*. Dalam teori himpunan *fuzzy* dimungkinkan untuk membuat derajat keanggotaan suatu objek dalam himpunan, sehingga dapat menyatakan peralihan keanggotaan suatu objek dalam himpunan. Peralihan ini secara bertahap dalam interval 0 dan 1 atau ditulis [0 1]. Hal ini sangat berkebalikan dengan himpunan klasik yang memiliki batasan tegas.[13]

Pengaplikasian FLC pada sistem *close loop* atau kalang tertutup terdiri atas masukan berupa *set point*, kontroler, *plant*, dan umpan balik. *Set point* merupakan nilai ideal atau nilai yang ingin dicapai dan sudah ditentukan pada awal. Lalu, pada bagian kontroler ada *error* dan *delta error* yang digunakan sebagai masukkan dari 12 proses awal, *error* merupakan selisih dari *set point* dengan umpan balik dan *delta error* merupakan selisih *error* dengan nilai *error* sebelumnya. Pengontrolan dengan *fuzzy* terdiri dari 4 bagian, yaitu fuzzifikasi (*fuzzification*), inferensi *fuzzy* (*fuzzy inference*) dimana penarikan kesimpulan akan didasari oleh basis aturan (*rule base*), dan defuzifikasi (*defuzzyfication*). Nilai keluaran dari sistem pengontrolan merupakan nilai keluaran yang tegas (*crisp*)[13]. Diagram pengendalian dengan FLC ditunjukkan oleh Gambar 2.7.



Gambar 2.7. Blok diagram sistem kontrol *closed loop* dengan logika *Fuzzy* [11]

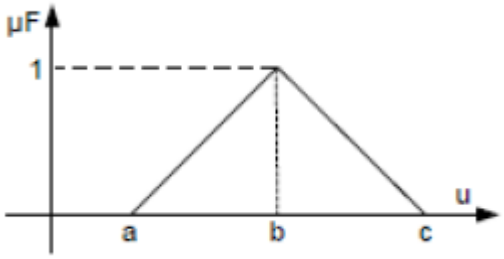
A. Fungsi Keanggotaan *Fuzzy*

Fungsi keanggotaan (*membership function)* merupakan suatu kurva yang menunjukkan pemetaan titik-titik *input* data ke dalam nilai keanggotaannya yang memiliki interval 0 sampai 1. Berikut adalah macam-macam fungsi keanggotaan *fuzzy* yang sering dipakai :

a. Fungsi Keanggotaan Segitiga

Kurva Segitiga pada dasarnya merupakan gabungan antara 2 garis yaitu linear naik dan linear turun. Garis linear naik adalah kenaikan himpunan dimulai dari nilai domain yang memiliki derajat keanggotaan nol (0) bergerak ke kanan menuju ke nilai domain yang memiliki derajat keanggotaan lebih tinggi sampai mencapai derajat keanggotaaan tertinggi (1) dan dilanjutkan garis linear turun yang bergerak menurun ke nilai domain yang

memiliki derajat keanggotaan lebih rendah [14]. Gambar 2.8 menunjukkan fungsi keanggotaan segitiga.



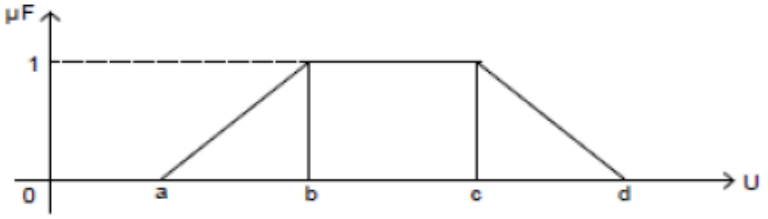
Gambar 2.8. Fungsi keanggotaan segitiga [12]

Fungsi keanggotaan segitiga mempunyai persamaan matematis 2.1 sebagai berikut:

$$\mu(u) = \begin{cases} 0 & ;untuk\ u \leq a \\ \left(\frac{u-a}{b-a}\right) & ;untuk\ a \leq u \leq b \\ 1 & ;untuk\ b \leq u \leq c \\ \left(\frac{c-u}{c-b}\right) & ;untuk\ u \geq c \\ 0 & \end{cases} \qquad (2.1)$$

b. Fungsi Keanggotaan Trapesium

Fungsi keanggotaan trapesium mirip seperti fungsi keanggotaan segitiga. Perbedaannya yaitu kurva trapesium memiliki beberapa titik yang nilai keanggotaannya 1. Fungsi keanggotaan trapesium ditunjukkan pada Gambar 2.9.



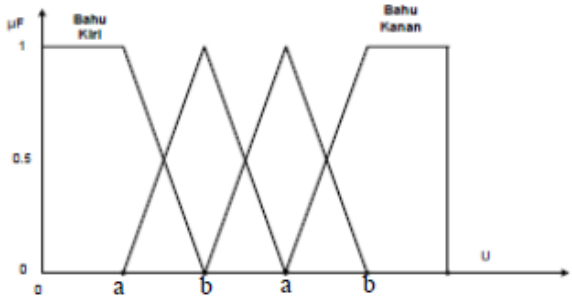
Gambar 2.9. Fungsi keanggotaan trapesium [13]

Fungsi keanggotaan trapesium mempunyai persamaan matematis 2.2 sebagai berikut:

$$\mu(u) = \begin{cases} 0 \\ \left(\frac{u-a}{b-a}\right) \\ 1 \\ \left(\frac{d-u}{d-c}\right) \\ 0 \end{cases} \begin{array}{l} ;untuk\ u \le a \\ ;untuk\ a \le u \le b \\ ;untuk\ b \le u \le c \\ ;untuk\ c \le u \le d \\ ;untuk\ u \ge d \end{array} \quad (2.2)$$

c. Fungsi Keanggotaan Bahu

Fungsi keanggotaan bentuk bahu pada dasarnya merupakan penggabungan antara fungsi keanggotaan bentuk segitiga dan trapezium. Daerah yang terletak ditengah merupakan bentuk segitiga, tetapi bagian paling kiri dan paling kanan tidak akan mengalami perubahan dan akan berbentuk seperti trapesium [15]. Himpunan *fuzzy* bahu digunakan untuk mengakhiri variabel suatu daerah *fuzzy*. Fungsi keanggotaan bentuk bahu dapat dilihat pada Gambar 2.10.



Gambar 2.10. Fungsi keanggotaan bentuk bahu [13]

Fungsi keanggotaan trapesium untuk bahu kiri mempunyai persamaan matematis 2.3 sebagai berikut:

$$\mu(u) = \begin{cases} 0 \\ \left(\frac{b-u}{b-1}\right) \\ 1 \end{cases} \begin{array}{l} ;untuk\ u \le a \\ :untuk\ a \le u \le b \\ ;untuk\ u \ge b \end{array} \quad (2.3)$$

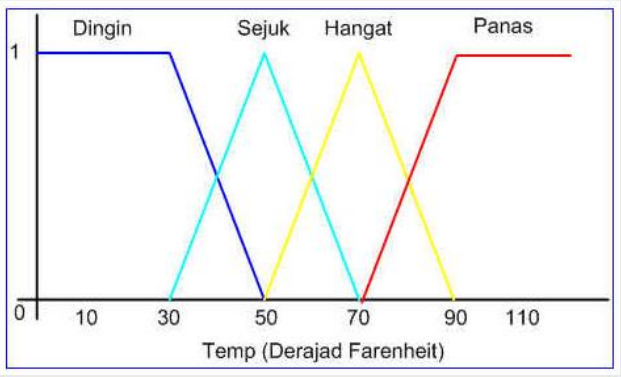
Sedangkan fungsi keanggotaan trapesium untuk bahu kanan mempunyai persamaan matematis 2.4 sebagai berikut:

$$\mu(u) = \begin{cases} 0 & ;untuk\ u \leq a \\ (\dfrac{u-a}{b-a}) & :untuk\ a \leq u \leq b \\ 1 & ;untuk\ u \geq b \end{cases} \quad (2.4)$$

B. Fuzzifikasi

Komponen fuzzifikasi berfungsi untuk memetakan masukan data tegas ke dalam himpunan *fuzzy* menjadi nilai derajat keanggotaan dengan melalui beberapa langkah, yaitu memperoleh nilai tegas dari variabel masukan, lalu memetakan nilai tegas tersebut ke dalam penyimpangan yang sesuai dengan himpunan *fuzzy* masukan.

Gambar 2.11 menunjukkan suatu contoh proses fuzzifikasi dimana nilai masukan tegas sebesar 50 dipetakan ke dalam fungsi keanggotaan segitiga dengan variabel linguistik *fuzzy* sebagai berikut: dingin, sejuh, hangat, dan panas.Variabel yang digunakan sebagai contoh adalah masukan nilai *temperature* 0 hingga 110.



Gambar 2.11. Proses fuzzifikasi

C. *Fuzzy Inference System*

Setelah nilai masukan didapat, terjadi proses pengambilan keputusan. Proses pengambilan keputusan pada *fuzzy* mengacu pada basis aturan yang ditentukan terlebih dahulu. Basis aturan berisi aturan-aturan kendali *fuzzy* yang digunakan untuk penentuan keputusan. Pembentukan basis data mencakup perancangan fungsi keanggotaan

untuk masing-masing variabel masukan dan keluaran, pendefinisian semesta pembicaraan dan penentuan variabel linguistik setiap variabel masukan dan keluaran [16].

Aturan pada suatu model *fuzzy* secara umum menggunakan aturan *IF-THEN*, dituliskan seperti persamaan 2.5.

$$if\ x\ is\ A\ then\ y\ is\ B \tag{2.5}$$

A dan B merupakan suatu nilai linguistik yang didefinisikan oleh suatu fungsi keanggotaan *fuzzy*. Pernyataan "$x\ is\ A$" merupakan pernyataan penyebab. Pernyataan "$y\ is\ B$" merupakan pernyataan akibat atau kesimpulan[16].

D. Inferensi *Fuzzy*

Proses inferensi *fuzzy* merupakan tahap evaluasi pada aturan fuzzy. Ada 3 jenis inferensi *fuzzy* yang sering digunakan dalam berbagai penelitian yaitu metode Mamdani, Tsukamoto dan Sugeno[16].

1. Metode Mamdani pertama kali diperkenalkan oleh Ibrahim Mamdani pada tahun 1975. Metode Mamdani menggunakan fungsi implikasi minimal dan agregasi maksimal sehingga metode ini bisa juga disebut dengan metode *MIN-MAX* [15]. Keluaran untuk didefinisikan dalam persamaan 2.6.

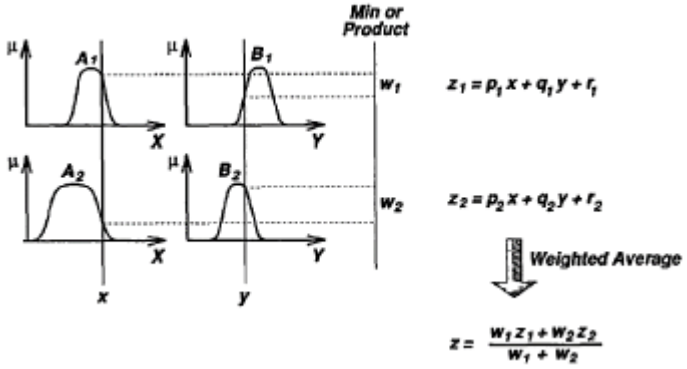
$$u_B k(y) = \max\left[\min\left[\mu A_1^k(x_i), \mu A_2^k(x_j)\right]\right]_k \tag{2.6}$$

2. Metode Tsukamoto merupakan metode dimana konsekuen yang berbentuk IF-Then direpresentasikan dengan fungsi keanggotaan yang monoton. *Output* yang dihasilkan adalah hasil inferensi dari tiap-tiap aturan yang diberikan secara tegas (*crisp)* dengan menggunakan rata-rata terbobot[17].
3. Metode Sugeno menghasilkan *output* berupa konstanta atau persamaan linier. Metode ini pertama kali dikenalkan oleh Sugeno Kang pada tahun 1985. Jika pada metode Mamdani proses defuzzifikasi menggunakan agregasi daerah kurva, maka pada metode Sugeno agregasi berupa *singleton-singleton.* Karakteristik dari *fuzzy* Sugeno

adalah bagian konsekuennya merupakan suatu persamaan linier dengan variabel-variabel sesuai dengan variabel-variabel masukannya. Ada 2 model untuk sistem inferensi *fuzzy* model Sugeno, yaitu model Sugeno orde-0 dan model orde-1. Model *Fuzzy* Sugeno orde-0 memiliki keluaran konstan yaitu z = k. Model *Fuzzy* Sugeno orde-1 memiliki keluaran z = p1*x1 + . . . + pn*xn+q dengan pi adalah suatu konstanta tegas ke-i dan q juga merupakan konstanta dalam konsekuen[18].

E. Defuzzifikasi

Gambar 2.12 merupakan merupakan tampilan dari inferensi *fuzzy* model Sugeno. Masukan yang digunakan merupakan himpunan *fuzzy* sedangkan bagian keluaran merupakan keluaran yang berupa persamaan linier.



Gambar 2.12. Inferensi *Fuzzy* model Sugeno [16]

Proses defuzzifikasi pada gambar 2.12 menggunakan metode *Weight of Average* (WOA) dengan rumus pada persamaan 2.7.

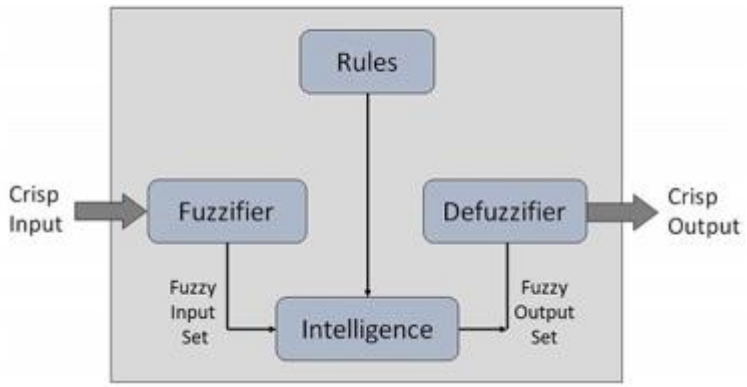
$$z = \frac{W_1.Z_1 + W_2.Z_2}{W_1 + W_2} \qquad (2.7)$$

Ada beberapa keuntungan dari metode Sugeno[18]:

1. Komputasinya lebih efisien
2. Bekerja paling baik untuk teknik linear (kontrol PID, dll) .
3. Bekerja paling baik untuk teknik optimasi dan adaptif .
4. Menjamin kontinuitas permukaan output .

5. Lebih cocok untuk analisis secara matematis.

   Fuzzy inference system (FIS) merupakan model nonlinear yang bersifat model *blackbox* yang dapat menjelaskan hubungan antara masukan dan keluaran dari suatu sistem *real* menggunakan suatu set aturan fuzzy. FIS memiliki hasil yang lebih akurat dibanding dengan model regresi linear yang konvensional. Logika fuzzy Sugeno dapat memodelkan berbagai macam sistem dari yang mudah hingga yang kompleks, struktur model dan mekanisme inferensinya bisa berubah-ubah tergantung masalah permodelannya. Metode ini dapat dianggap sebagai suatu solusi yang fleksibel [19].

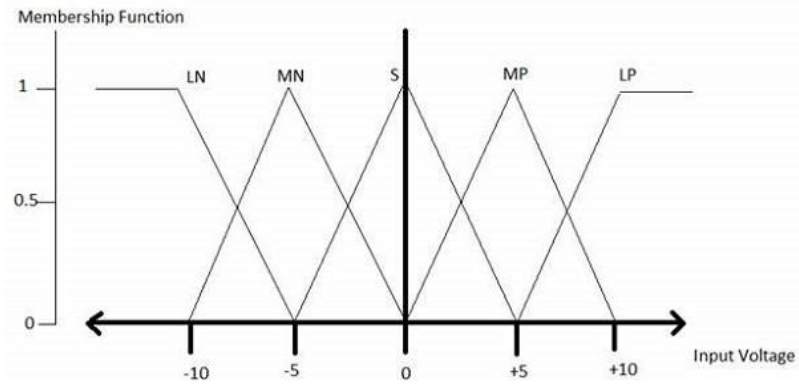


Gambar 2.13. *Fuzzy Logic Control*

Pertama kali dikembangkan oleh Dr. Lotfi Zadeh di University of California pada tahun 1960an, ketika sedang mencoba menyelesaikan sebuah permasalahan – agar komputer dapat mengerti dan berfikir selayaknya manusia, dengan "bahasa natural". Dr. Zadeh menyadari bahwa diantara "Ya" dan "Tidak", masih terdapat range dari kemungkinan-kemungkinan yang mungkin ada, dan boolean logic (logika biner yang selama ini kita kenal) yang hanya dapat bernilai "Ya" dan "Tidak" tidaklah mencukupi.

1. *Rules*: set peraturan yang berisikan kondisi-kondisi dan apa yang dilakukan jika kondisi terpenuhi.

2. *Intelligence*: seperti namanya, *intelligence* mensimulasikan proses pemikiran manusia, berdasarkan pencocokan fuzzy *input set* dan *rules* yang ada, dan menghasilkan *fuzzy output set*.
3. *Fuzzifier*: merubah input *crisp* menjadi fuzzy *input set* melalui sebuah proses yang disebut fuzifikasi
4. *Defuzzifier*: merubah fuzzy *output set* menjadi crisp output melalui proses defuzifikasi, menghasilkan output final dari system



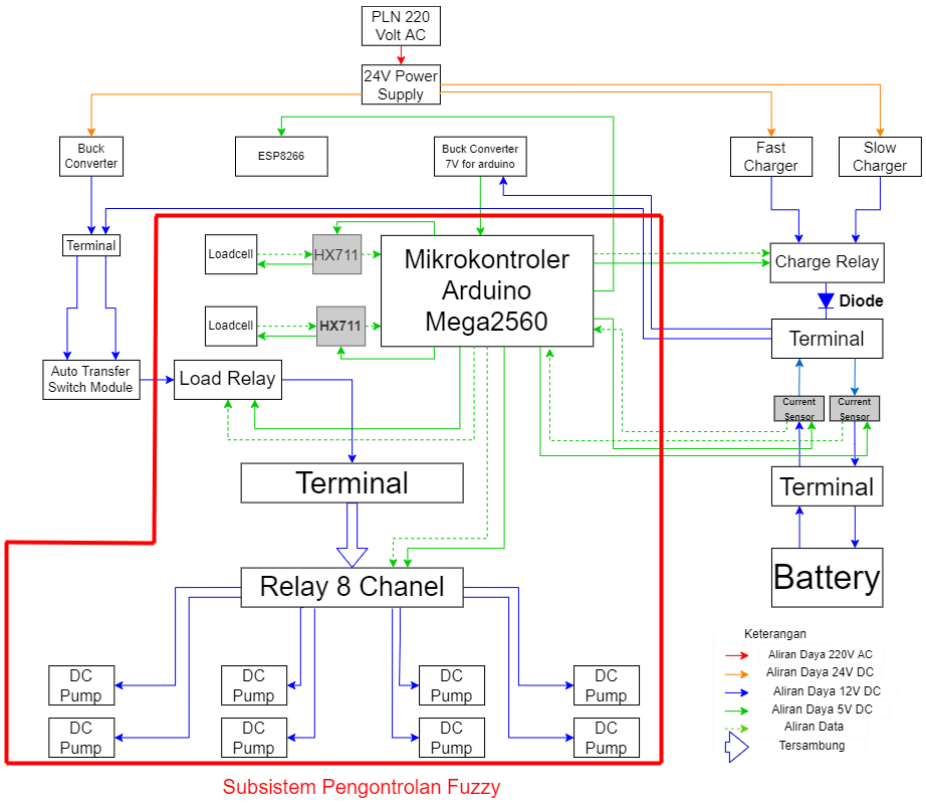
Gambar 2.14. Proses fuzzifikasi

Dalam proses fuzifikasi ataupun defuzifikasi, salah satu metoda yang umum adalah dengan pemberian *membership function* ke input. Dengan *membership function*, hal-hal linguistik dari input (jenis-jenis kemungkinan yang dapat terjadi) dapat dipetakan menjadi fuzzy *set* secara grafis dalam *range* 0-1. Tipe fuzifikasi yang paling umum digunakanan adalah fuzifikasi *triangular* (bentuk dari grafik fuzzy *set*). Dalam grafik, *input space* atau sumbu x biasa disebut sebagai *universe of discourse* atau *universal set* (u). Pada contoh fuzifikasi *triangular* berikut, *universal set* berisikan nilai-nilai tegangan *input* (Volt) yang mungkin, yang berkisar dari -10 ~ 10 V. Sumbu y merupakan *membership value* dalam range 0~1.

BAB III
PERANCANGAN SISTEM

Perancangan Tugas Akhir ini pada dasarnya dibagi menjadi dua bagian, yaitu perancangan perangkat keras (*hardware*) dan perancangan program (*software*). Perancangan perangkat keras terdiri dari blok diagram perangkat keras, rangkaian mikrokontroler Arduino ATMega 2560 Pro Mini, modul relai, modul HX711, Pompa air DC 12V, dan *loadcell*. Perancangan program berupa sesuai bahasa perintah tersebut dalam bahasa mesin yang membuat system dapat bekerja sesuai bahasa perintah tersebut yang meliputi program fitur aplikasi android.

Parameter keberhasilan pada perancangan Tugas Akhir ini adalah sistem dapat menampilkan data pemantauan melalui aplikasi Android secara langsung. Gambar 3.1 menjelaskan diagram blok perancangan purwarupa secara keseluruhan. Untuk Tugas Akhir Perancangan MCU Pada Pengaturan *Volume* Cairan Menggunakan *Fuzzy Logic Controler* Metode Sugeno yang ditandai dengan kotak berwarna merah. Menggunakan beberapa komponen antara lain ; 1 buah mikrokontroler Arduino ATMega2560 Promini, 2 buah sensor *loadcell* beserta modul penguatnya yaitunya HX-711, 1 buah load relai, 1 buah terminal, 2 buah modul relai 12 v dengan masing – masing 4 chanel, dan 8 buah Pompa air dc 12v. Pada sistem perancangan kontrol ini berfungsi untuk menyuplai cairan keseluruhan mulai dari tabung – tabung bahan hingga pengguna mendapatkan cairan pilihan mereka berupa sanitasi ataupun desinfektan.

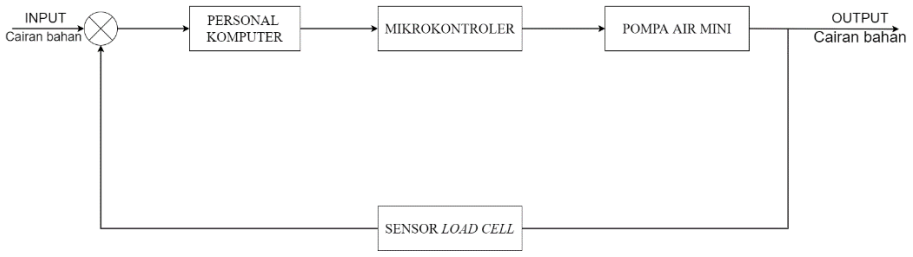


Gambar 3.1. Diagram purwarupa sistem dispenser otomatis

3.1 Perancangan Sistem

Perancangan *prototype* kontroler Tugas Akhir ini berbasis logika *fuzzy* yang menggunakan sistem kontrol kalang terbuka. Kontroler yang dirancang pada Tugas Akhir ini digunakan untuk menyesuaikan keputusan dari simulasi *fuzzy* yang sudah disusun sesuai dengan perintah *user*/pengguna. Adapula sistem kontrol on/off untuk menutup dan membuka pompa air DC 12V yang hanya disesuaikan berupa detik menggunakan *coding* Arduino. Perancangan ini menggunakan modul miktrokontroler Arduino ATMega 2560 Pro Mini yang berperan sebagai tempat untuk menyimpan data kontroler, pembacaan sensor, pengontrol rangkaian elektronik seperti sensor *load cell*, sensor *relay*, pompa air DC 12V, serta monitoring yang menggunakan modul ESP8266. Penggunaan miktrokontroler Arduino ATMega 2560 Pro Mini pada perancangan ini mampu memproses data keseluruhan yang sudah terintegrasi dengan penggunaan memori sebesar 80% secara keseluruhan. Sensor *load cell* merupakan sensor pembacaan berat atau bisa

disebut sensor timbangan yang memiliki sensivitas terhadap berat suatu barang. Sensor *load cell* pada perancangan ini berfungsi untuk membaca hasil berat dari tabung *mixer/buffer* sanitiasi dan desinfektan berupa volume yang apakah sudah sesuai dengan banyaknya cairan dalam tabung. Perancangan ini menggunakan 9 pompa air DC 12V, yang dimana masing masing valve ini memiliki fungsinya sendiri sendiri sesuai dengan kebutuhan alat. Dimana untuk tabung *buffer/mixer* untuk sanitasi membutuhkan 4 pompa air DC 12V, yaitu 1 pompa air DC 12V untuk bahan glycerin, 1 pompa air DC 12V untuk bahan H_2O_2, 1 pompa air DC 12V untuk bahan alcohol, kemudian 1 Pompa air DC 12V untuk bukaan yang akan diteruskan kepada tabung *user/*pengguna. Untuk tabung *buffer/mixer* desinfektan membutuhkan 5 pompa air DC 12V yang terdiri dari : 1 pompa air DC 12V bahan bayclin, 1 pompa air DC 12V bahan wipol, 1 pompa air DC 12V bahan Dettol, 1 pompa air DC 12V bahan alcohol, dan 1 pompa air DC 12V untuk bukaan yang akan diteruskan kepada tabung *user/*pengguna. Sensor *load cell* ini berfungsi juga sebagai *trigger* dimana ketika isian tangki *buffer* sanitasi ataupun disinfektan sudah mencapai batas maksimum, kemudian system tangki bahan akan bekerja untuk memulai pengisian dalam *buffer*. Sensor *load cell* ini membutuhkan sebuah modul yang berfungsi untuk pembaca berat pada sensor berat *(load cell)* dalam pengukuran berat yaitu modul HX-711. Gambar 3.2 merupakan ilustrasi diagram blok sistem kontrol secara keseluruhan.



Gambar 3.2. Blok diagram sistem proses pengisian tangki *buffer*

Berdasarkan pada Gambar 3.2 dapat diketahui bahwa sistem akan bekerja dengan memberikan masukan berupa *set point* yang sudah ditentukan yaitu cairan – cairan pada tiap tabung bahan. Selanjutnya, akan diteruskan atau diolah oleh

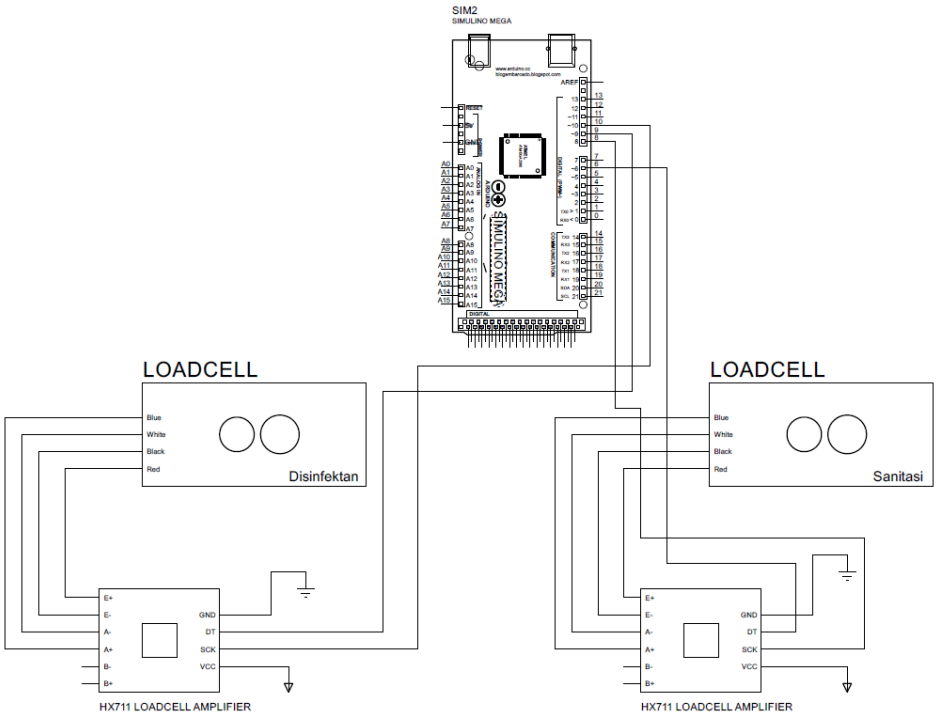
miktrokontroler yang bekerja untuk mengatur batasan minimal dan maksimal pada tabung *buffer* atau *mixer* dan aktuator Pompa air DC 12V bekerja pada sistem ON/OFF. Setelah itu, sensor *load cell* akan membaca dan memberikan umpan balik ke sistem apakah sudah sesuai dengan *volume.*

3.2 Perancangan Perangkat Keras

Pada sub bab ini dibahasa diagram blok dan perancangan perangkat keras pada sistem kontroler alat dispender otomatis menggunakan metode logika *Fuzzy* Sugeno pada tugas akhir ini terdiri dari modul mikrokontroler yang terhubung dengan beberapa aktuator pendukung seperti pompa air DC 12V dan sensor *load cell*.

3.2.1 Perangkat Keras Sensor *Load Cell*

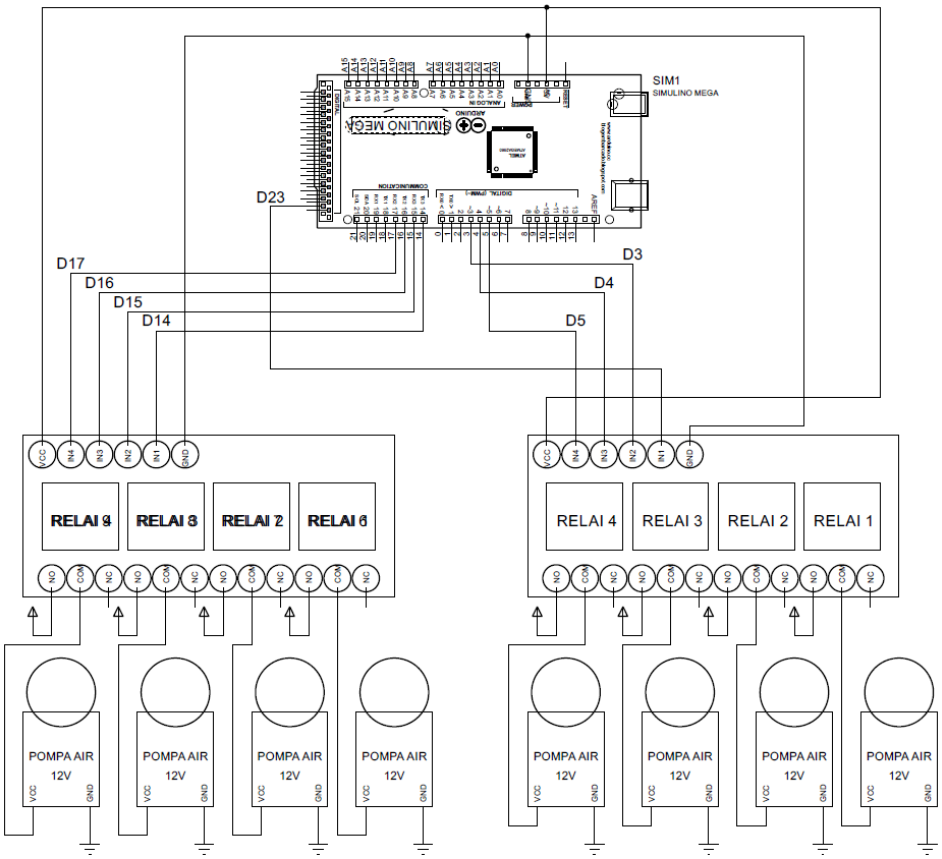
Pada Gambar 3.3 menunjukkan perangkat keras dimana sensor *load cell* yang terhubung dengan Mikrokontroler Arduino ATMega2560 Pro Mini. Pada sensor *load cell* terdapat 4 pin warna *blue*, *white*, *black*, dan *red* yang terhubung dengan kaki modul HX711 sebagai penguat sensor tersebut E+ (*blue*), E- (*white*), A- (*black*), dan A+ (*red*). Kemudian VCC dan GND terhubung pada *ground* dan sumber untuk kedua sensor *load cell*. Perancangan alat dispenser otomatis ini menggunakan sensor *load cell* yang dihubungkan pada PORT DIGITAL 6 (DT) dan 8 (SCK) untuk *load cell* hand sanitasi, sedangkan sensor *load cell* untuk disinfektan dihubungkan pada PORT DIGITAL 9 (DT) dan 10 (SCK).



Gambar 3.3. Perangkat keras dari *prototype* sensor *loadcell*

3.2.2 Perangkat Keras Penyusunan Arduino ATMega2560 Promini dengan Relai

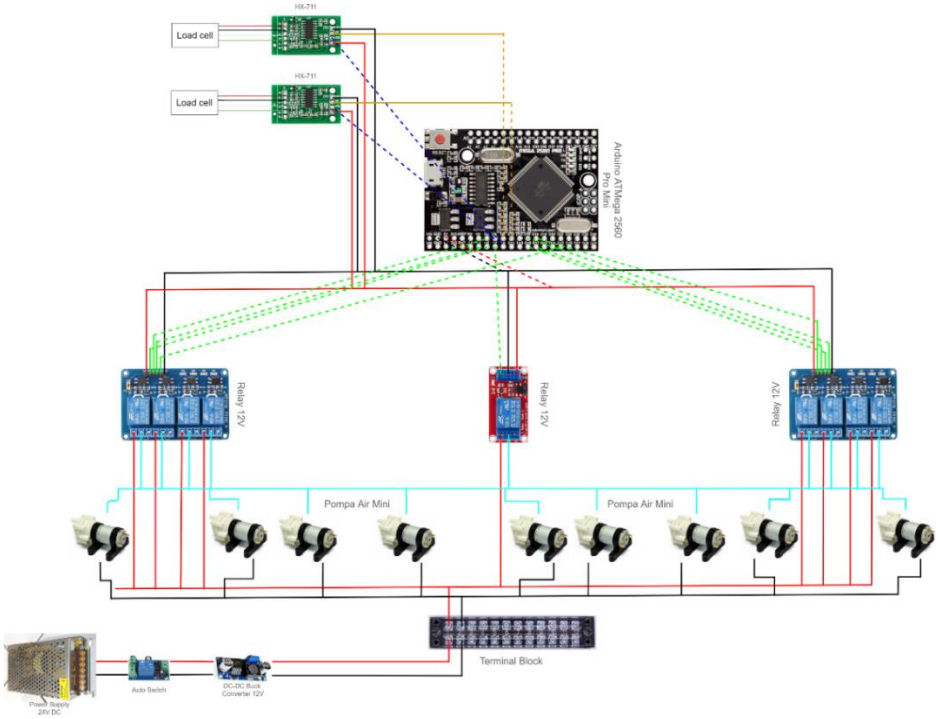
Pada Gambar 3.4 merupakan rangkaian relai interkoneksi dengan mikrokontroler Arduino ATMega 2560 Promini perancangan alat dispenser otomatis. Modul relai ini bekerja dengan mendapatkan *input* PORT DIGITAL dari Arduino ATMega yang dimana pin COM pada relai terhubung dengan aktuator (pompa air DC 12v). Untuk alokasi PORT DIGITAL relai terhubung pada pin 23 (relai 1), pin 3 (relai 2), pin 4 (relai 3), pin 5 (relai 4), pin 14 (relai 6), pin 15 (relai 7), pin 16 (relai 8), dan pin 17 (relai 9), sesuai dengan gambar berikut.



Gambar 3.4. Perangkat keras dari *prototype* modul relai dan aktuator

3.2.3 Perancangan Mikrokontroller Arduino ATMega 2560 Pro Mini

Mikrokontroler berfungsi sebagai pengendali utama pada perangkat prototipe alat yang digunakan pada penelitian ini. Mikrokontroler yang digunakan adalah mikrokontroler Arduino ATMega 2560 Pro Mini. Kontrol aktuator meliputi kontrol Pompa air DC 12V yang disambung seperti pada Gambar 3.5 yang merupakan diagram blok skema secara keseluruhan pada perancangan perangkat keras sistem Perancangan Mesin Pencampur Cairan Sanitasi dan Disinfektan.



Gambar 3.5. Alokasi Port mikrokontroler Arduino ATMega2560 Pro Mini

Penjelasan dari bagian – bagian yang terdapat pada modul mikrokontroler adalah sebagai berikut :

1. Sensor *Load Cell* sebagai sensor berat / timbangan untuk pengukuran *volume* tabung *buffer/mixer* sanitasi maupun disinfektan.
2. Modul HX711 merupakan modul *amplifier* (penguat sinyal) sekaligus modul analog to digital converter untuk mengondisikan sinyal analog dari sensor *load cell* sekaligus untuk mengkonversikannya menjadi sinyal digital.
3. Mikrokontroler Arduino ATMega 2560 Pro Mini sebagai pusat pengendali.
4. Modul relai 12V sebagai saklar otomatis yang dikontrol melalui program Arduino.
5. Pompa air DC 12V merupakan aktuator yang akan mengontrol buka tutupnya cairan otomatis.

3.3 Perancangan Metode Kontroler *Fuzzy Logic Controller* (FLC)

Pada penelitian ini menggunakan metode kontrol logika *Fuzzy Logic Controller* (FLC). Masukan dari sistem kontrol berasal dari pembacaan sensor *load cell* dan *set point.* Model FLC yang digunakan dalam penelitian kali ini adalah model Sugeno. Model Sugeno mempunyai rumus untuk Defuzzifikasi. Untuk aturan IFTHEN *Fuzzy* dalam persamaan RU(k)= IF x1 is $A1k$ and… and xn is Ank THEN y is Bk, dimana $A1k$ dan Bk berturut-turut adalah himpunan Fuzzy dalam Ui R (U dan V adalah domain fisik), i = 1,2, …, n dan x = ($x1$, $x2$, …, xn) U dan y V berturut-turut adalah variabel input dan output (linguistik) dari sistem *Fuzzy*. Defuzzifier pada persamaan di atas didefinisikan sebagai suatu pemetaan dari himpunan *Fuzzy* B ke dalam V R (yang merupakan output dari inferensi *Fuzzy*) ke titik tegas $y*V$. Pada metode Sugeno *defuzzification* dilakukan dengan perhitungan *Weight Average* (*WA*) seperti pada persamaan 3.1 [20]:

$$WA = \frac{a_{1z_1} + a_{2z_2} + a_{3z_3} + \cdots + a_{nz_n}}{a_1 + a_2 + a_3 + \cdots + a_n} \quad (3.1)$$

Keterangan :

WA = Nilai rata – rata

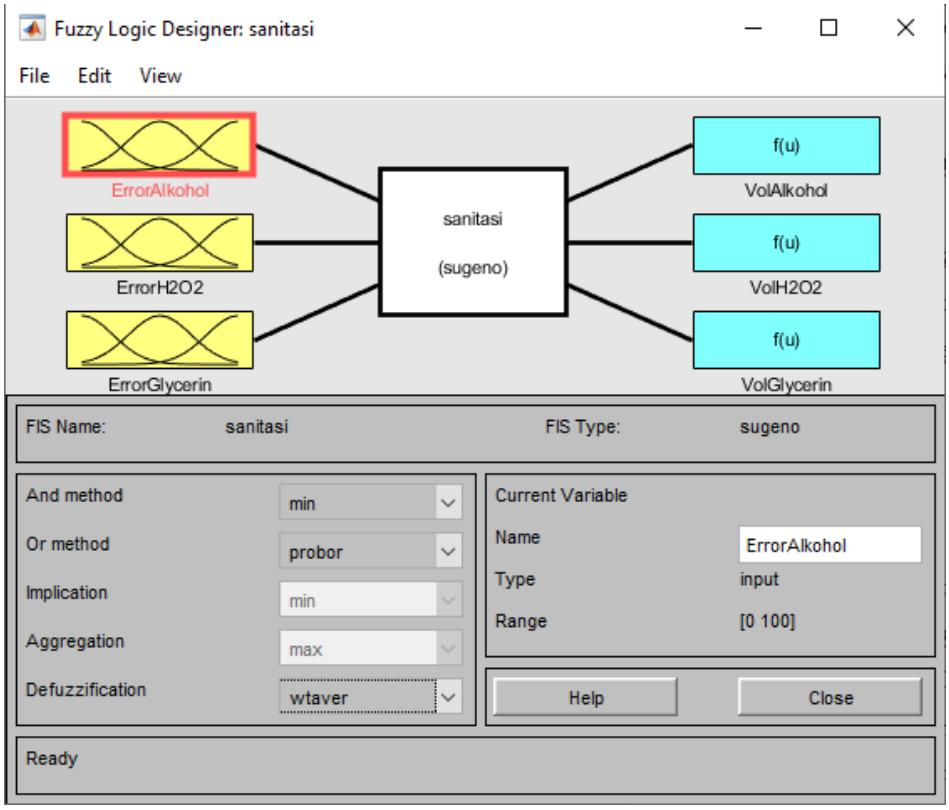
a_n = Nilai predikat aturan ke – *n*

z_n = Indeks nilai output (konstanta) ke – *n*

3.3.1 Perancangan *Fuzzy* Pada Proses Sanitasi

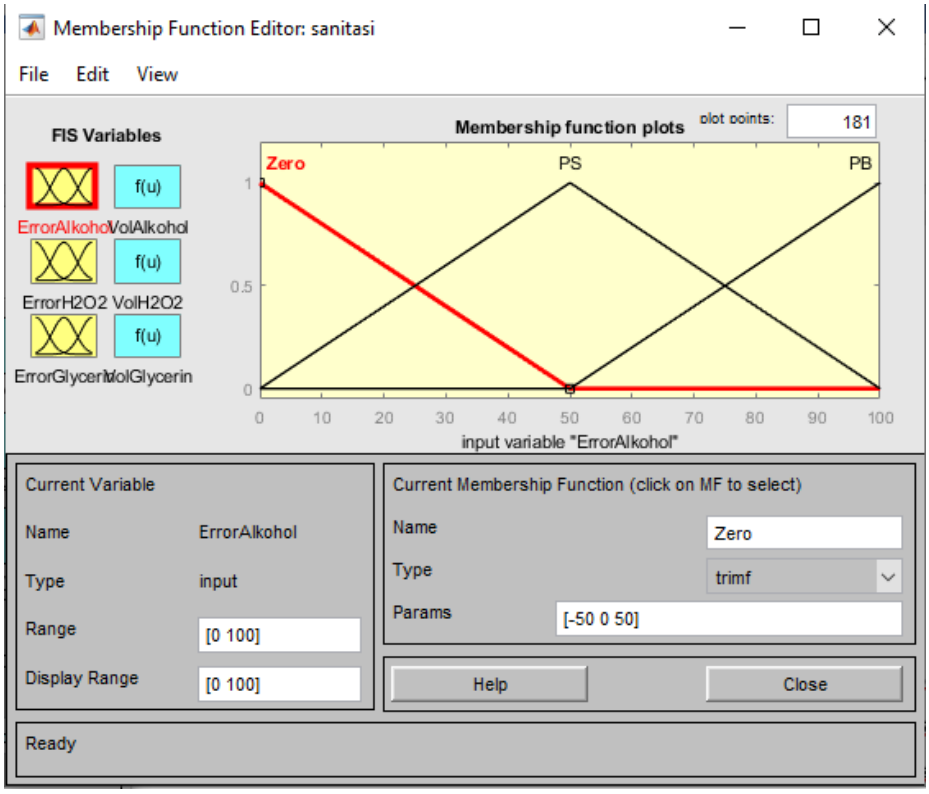
Perancangan *Fuzzy* pada alat dispenser otomatis ini menggunakan data *error* volume tiap bahan untuk tangki penyimpan sanitasi ataupun disinfektan sebagai *input* dan presentasi volume tiap bahan sebagai *output* yang berfungsi untuk mengendalikan kepresisian ml dalam bentuk *volume*. Metode *Fuzzy* yang digunakan dalam kontroler ini adalah metode *Fuzzy* Sugeno, dimana metode *AND* yang digunakan adalah nilai minimal, metode *OR* yang digunakan adalah nilai maksimal, dan tipe *defuzzification* yang digunakan adalah *weight of average* atau wtaver. Kemudian pada proses sanitasi menggunakan 3 bahan olahan awal sebelum menjadi hasil yang diinginkan oleh pengguna/*user* terdapat bahan – bahan Alkohol (25%), H_2O_2 (50%), dan Glycerin (25%). Proses *Fuzzy Logic* menggunakan metode

Sugeno pada Tugas Akhir ini, terdapat 3 *input* dan 3 *output* yang sudah disesuaikan dengan peneliti dan didapatkan hasil akhir untuk *rule base*nya sebanyak 27 *rules* pada kedua proses akhir yaitu cairan sanitasi dan disinfektan. Gambar 3.6 menunjukkan *Fuzzy Logic Designer* untuk tabung sanitasi.



Gambar 3.6. *Fuzzy Logic Designer* sanitasi

Pada *input error* ketiga bahan sanitasi terdapat 3 fungsi keanggotaan yaitu *Zero, PS (*Positif *Small)*, dan PB (Positif *Big*). Pada masing – masing *input*an memiliki *domain range* atau batasan nilai yang sesuai dengan data dan sudah dilakukan oleh peneliti. Untuk lebih jelasnya bisa dilihat pada Gambar 3.7 yang dimana menunjukkan kurva *domain range* atau batasan nilai sebagai contoh pada *input error* Alkohol sanitasi.



Gambar 3.7. *Membership function input error* Alkohol sanitasi

Tabel 3.1 berikut merupakan batasan nilai untuk *input error* Alkohol sanitasi.

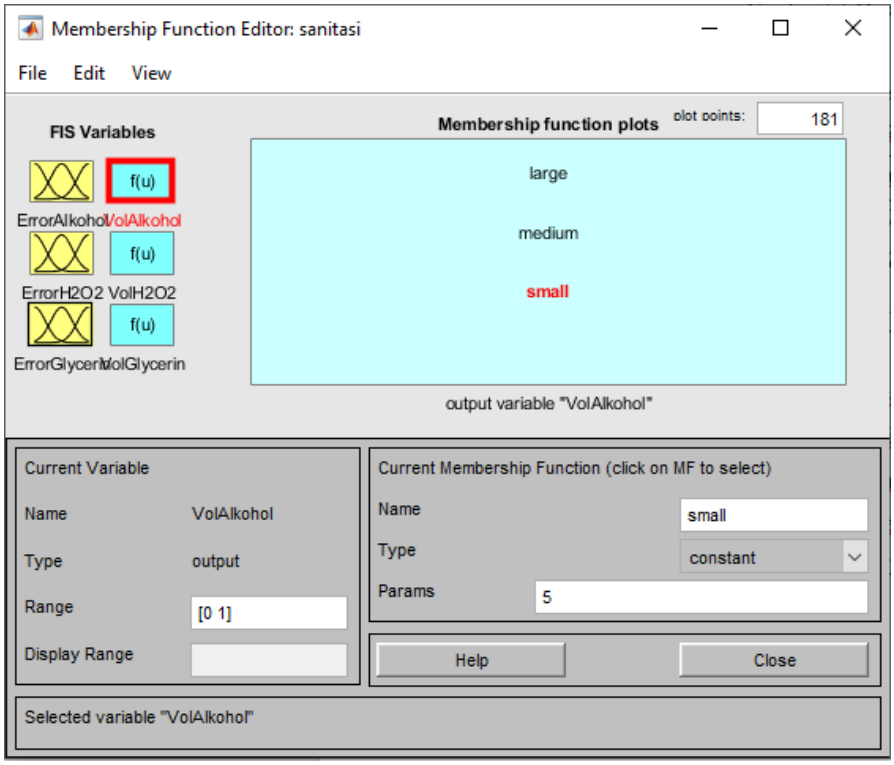
Tabel 3.1. Domain range input error Alkohol sanitasi

No.	Fungsi Keanggotaan	*Range* (%)
1	*Zero*	[0 ≤50]
2	Positif *Small* (PS)	[0 ≤100]
3	Positif *Big* (PB)	[50 ≥100]

Untuk kedua bahan mengolah sanitasi H_2O_2 dengan *Glycerin* sama seperti kurva *domain range* Alkohol beserta dengan isi tabel.

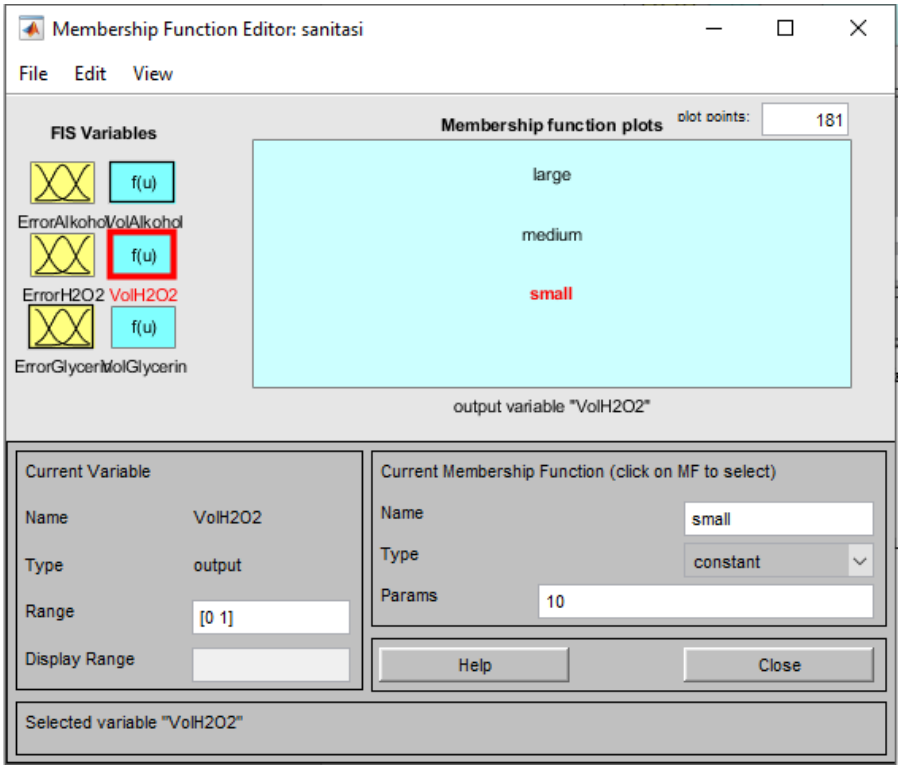
Pada perancangan kontroler ini selain memiliki 3 *input error*, juga memiliki 3 *output* yang terdiri dari 3 persentase volume, yaitu *volume* alkohol untuk sanitasi, *volume* H_2O_2, dan *volume* glycerin. Berikut ini merupakan *membership function* dari ketiga set *output* yang dihasilkan yaitu *volume* alkohol untuk sanitasi yang dapat

dilihat pada Gambar 3.8, *volume* H_2O_2 yang dapat dilihat pada Gambar 3.9, dan *volume* glycerin yang dapat dilihat pada Gambar 3.10.



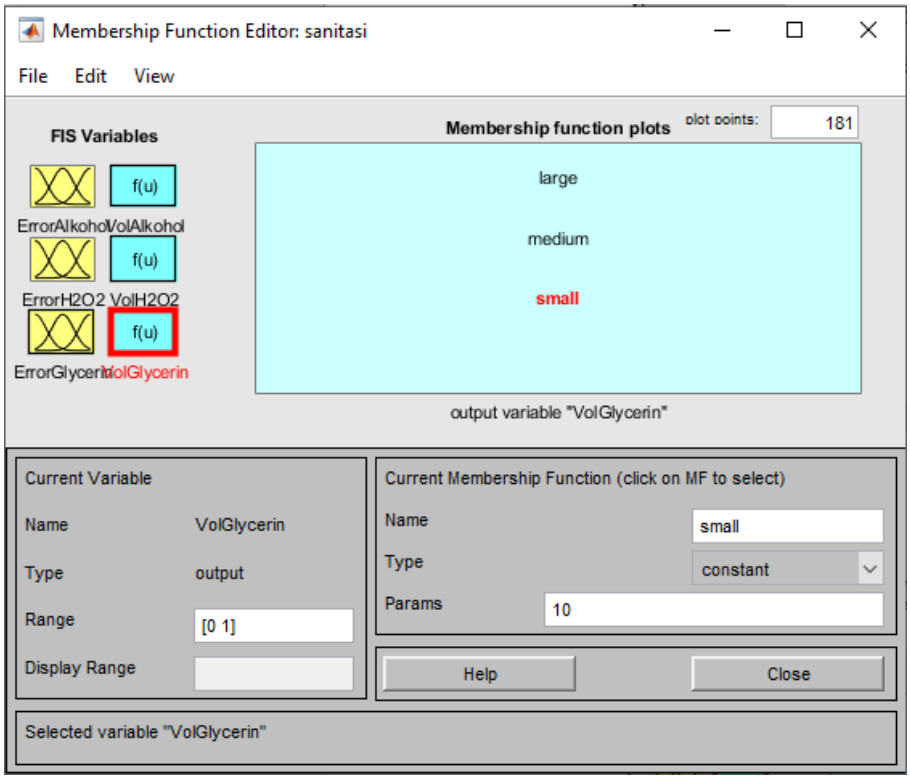
Gambar 3.8. *Membership function output* volume alkohol sanitasi

Gambar 3.8 menjelaskan *membership function* keluaran saat volume alkohol sanitasi *small*.



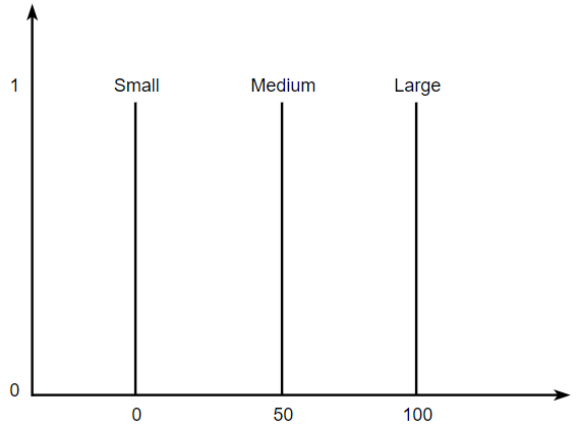
Gambar 3.9. *Membership function output* volume H_2O_2

Gambar 3.9 menjelaskan *membership function* keluaran saat volume H_2O_2 *small.*

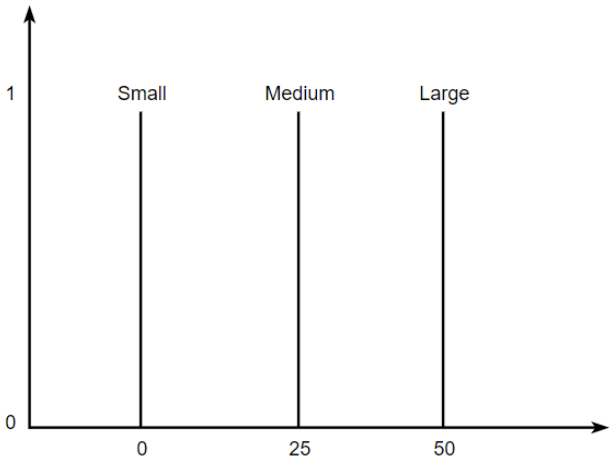


Gambar 3.10. *Membership function output* volume glycerin

Pada keluaran *Fuzzy* dalam perancangan ini memiliki keluaran berupa *singleton* yang sama untuk *output* H_2O_2 dan Glycerin dengan jangkauan *range* 0 sampai dengan 100, begitupula untuk hasil *output* alkohol sanitasi dengan jangkauan *range* 0 sampai dengan 50 menjadi 3 variabel lingustik. Variabel untuk *output* H_2O_2 dan Glycerin, Small dengan nilai *singleton* 0, Medium dengan nilai *singleton* 50, Large dengan nilai *singleton* 100. Sedangkan untuk variabel *output* alkohol, Small dengan nilai *singleton* 0, Medium dengan nilai *singleton* 25, Large dengan nilai *singleton* 50. Berikut pada Gambar 3.11 merupakan nilai *singleton output* H_2O_2 dan Glycerin yang digunakan dan pada Gambar 3.12 merupakan nilai *singleton output* alkohol sanitasi yang digunakan.



Gambar 3.11. *Membership function output* H_2O_2 dan Glycerin



Gambar 3.12. *Membership function output* Alkohol sanitasi

Rule base yang diperoleh dari perancangan kontroler ini berjumlah 27 *rules* yang dapat dilihat sebagai berikut :

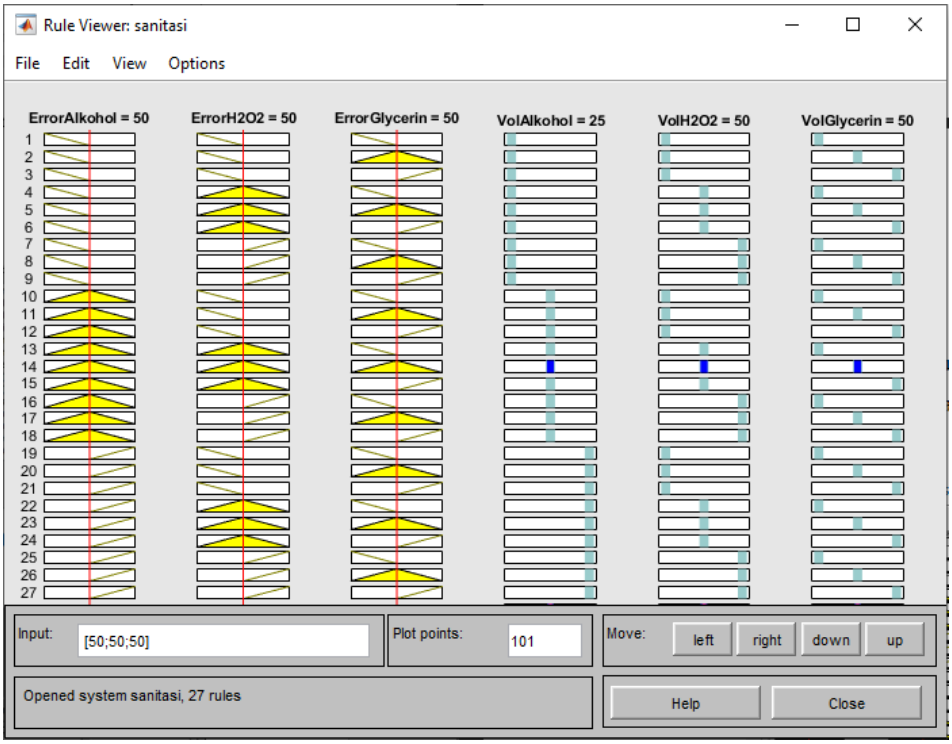
1. Jika *Error Alkohol* adalah *Zero* && *Error* H_2O_2 adalah *Zero* && *Error Glycerin* adalah *Zero* maka Volume *Alkohol* adalah Small && Volume H_2O_2 adalah Small && Volume *Glycerin* adalah Small.
2. Jika *Error Alkohol* adalah *Zero* && *Error* H_2O_2 adalah *Zero* && *Error Glycerin* adalah PS maka Volume *Alkohol* adalah Small && Volume H_2O_2 adalah Small && Volume *Glycerin* adalah Medium.

3. Jika *Error Alkohol* adalah *Zero* && *Error* H_2O_2 adalah *Zero* && *Error Glycerin* adalah PB maka Volume *Alkohol* adalah Small && Volume H_2O_2 adalah Small && Volume *Glycerin* adalah Large.
4. Jika *Error Alkohol* adalah *Zero* && *Error* H_2O_2 adalah PS && *Error Glycerin* adalah *Zero* maka Volume *Alkohol* adalah Small && Volume H_2O_2 adalah Medium && Volume *Glycerin* adalah Small.
5. Jika *Error Alkohol* adalah *Zero* && *Error* H_2O_2 adalah PS && *Error Glycerin* adalah PS maka Volume *Alkohol* adalah Small && Volume H_2O_2 adalah Medium && Volume *Glycerin* adalah Medium.
6. Jika *Error Alkohol* adalah *Zero* && *Error* H_2O_2 adalah PS && *Error Glycerin* adalah PB maka Volume *Alkohol* adalah Small && Volume H_2O_2 adalah Medium && Volume *Glycerin* adalah Large.
7. Jika *Error Alkohol* adalah *Zero* && *Error* H_2O_2 adalah PB && *Error Glycerin* adalah *Zero* maka Volume *Alkohol* adalah Small && Volume H_2O_2 adalah Small && Volume *Glycerin* adalah Large.
8. Jika *Error Alkohol* adalah *Zero* && *Error* H_2O_2 adalah PB && *Error Glycerin* adalah PS maka Volume *Alkohol* adalah Small && Volume H_2O_2 adalah Medium && Volume *Glycerin* adalah Large.
9. Jika *Error Alkohol* adalah *Zero* && *Error* H_2O_2 adalah PB && *Error Glycerin* adalah PB maka Volume *Alkohol* adalah Small && Volume H_2O_2 adalah Large && Volume *Glycerin* adalah Large.
10. Jika *Error Alkohol* adalah PS && *Error* H_2O_2 adalah *Zero* && *Error Glycerin* adalah *Zero* maka Volume *Alkohol* adalah Medium && Volume H_2O_2 adalah Small && Volume *Glycerin* adalah Small.
11. Jika *Error Alkohol* adalah PS && *Error* H_2O_2 adalah *Zero* && *Error Glycerin* adalah PS maka Volume *Alkohol* adalah Medium && Volume H_2O_2 adalah Small && Volume *Glycerin* adalah Medium.
12. Jika *Error Alkohol* adalah PS && *Error* H_2O_2 adalah *Zero* && *Error Glycerin* adalah PB maka Volume *Alkohol* adalah Medium && Volume H_2O_2 adalah Small && Volume *Glycerin* adalah Large.

13. Jika *Error Alkohol* adalah PS && *Error* H_2O_2 adalah PS && *Error Glycerin* adalah *Zero* maka Volume *Alkohol* adalah Medium && Volume H_2O_2 adalah Medium && Volume *Glycerin* adalah Small.
14. Jika *Error Alkohol* adalah PS && *Error* H_2O_2 adalah PS && *Error Glycerin* adalah PS maka Volume *Alkohol* adalah Medium && Volume H_2O_2 adalah Medium && Volume *Glycerin* adalah Medium.
15. Jika *Error Alkohol* adalah PS && *Error* H_2O_2 adalah PS && *Error Glycerin* adalah PB maka Volume *Alkohol* adalah Medium && Volume H_2O_2 adalah Medium && Volume *Glycerin* adalah Large.
16. Jika *Error Alkohol* adalah PS && *Error* H_2O_2 adalah PB && *Error Glycerin* adalah *Zero* maka Volume *Alkohol* adalah Medium && Volume H_2O_2 adalah Large && Volume *Glycerin* adalah Small.
17. Jika *Error Alkohol* adalah PS && *Error* H_2O_2 adalah PB && *Error Glycerin* adalah PS maka Volume *Alkohol* adalah Medium && Volume H_2O_2 adalah Large && Volume *Glycerin* adalah Medium.
18. Jika *Error Alkohol* adalah PS && *Error* H_2O_2 adalah PB && *Error Glycerin* adalah PB maka Volume *Alkohol* adalah Medium && Volume H_2O_2 adalah Large && Volume *Glycerin* adalah Large.
19. Jika *Error Alkohol* adalah PB && *Error* H_2O_2 adalah *Zero* && *Error Glycerin* adalah *Zero* maka Volume *Alkohol* adalah Large && Volume H_2O_2 adalah Small && Volume *Glycerin* adalah Small.
20. Jika *Error Alkohol* adalah PB && *Error* H_2O_2 adalah *Zero* && *Error Glycerin* adalah PS maka Volume *Alkohol* adalah Large && Volume H_2O_2 adalah Small && Volume *Glycerin* adalah Medium.
21. Jika *Error Alkohol* adalah PB && *Error* H_2O_2 adalah *Zero* && *Error Glycerin* adalah PB maka Volume *Alkohol* adalah Large && Volume H_2O_2 adalah Small && Volume *Glycerin* adalah Large.
22. Jika *Error Alkohol* adalah PB && *Error* H_2O_2 adalah PS && *Error Glycerin* adalah *Zero* maka Volume *Alkohol* adalah Large && Volume H_2O_2 adalah Medium && Volume *Glycerin* adalah Small.

23. Jika *Error Alkohol* adalah PB && *Error* H_2O_2 adalah PS && *Error Glycerin* adalah PS maka Volume *Alkohol* adalah Large && Volume H_2O_2 adalah Medium && Volume *Glycerin* adalah Medium.
24. Jika *Error Alkohol* adalah PB && *Error* H_2O_2 adalah PS && *Error Glycerin* adalah PB maka Volume *Alkohol* adalah Large && Volume H_2O_2 adalah Medium && Volume *Glycerin* adalah Large.
25. Jika *Error Alkohol* adalah PB && *Error* H_2O_2 adalah PB && *Error Glycerin* adalah *Zero* maka Volume *Alkohol* adalah Large && Volume H_2O_2 adalah Large && Volume *Glycerin* adalah Small.
26. Jika *Error Alkohol* adalah PB && *Error* H_2O_2 adalah PB && *Error Glycerin* adalah PS maka Volume *Alkohol* adalah Large && Volume H_2O_2 adalah Large && Volume *Glycerin* adalah Medium.
27. Jika *Error Alkohol* adalah PB && *Error* H_2O_2 adalah PB && *Error Glycerin* adalah PB maka Volume *Alkohol* adalah Large && Volume H_2O_2 adalah Large && Volume *Glycerin* adalah Large.

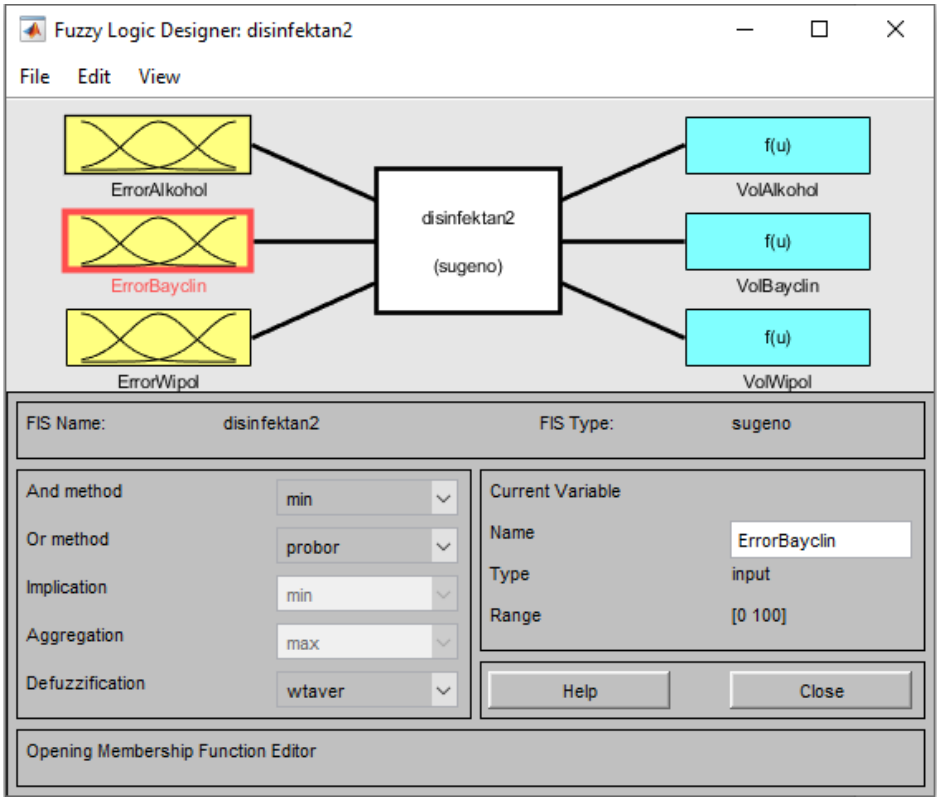
Pada Gambar 3.13 menunjukkan *rule viewer* sanitasi pada aplikasi simulasi MATLAB yang dapat disimulasikan menggunakan logika *Fuzzy* dengan tiga masukkan dan tiga keluaran nilai sebagai berikut.



Gambar 3.13. *Rule viewer* sanitasi

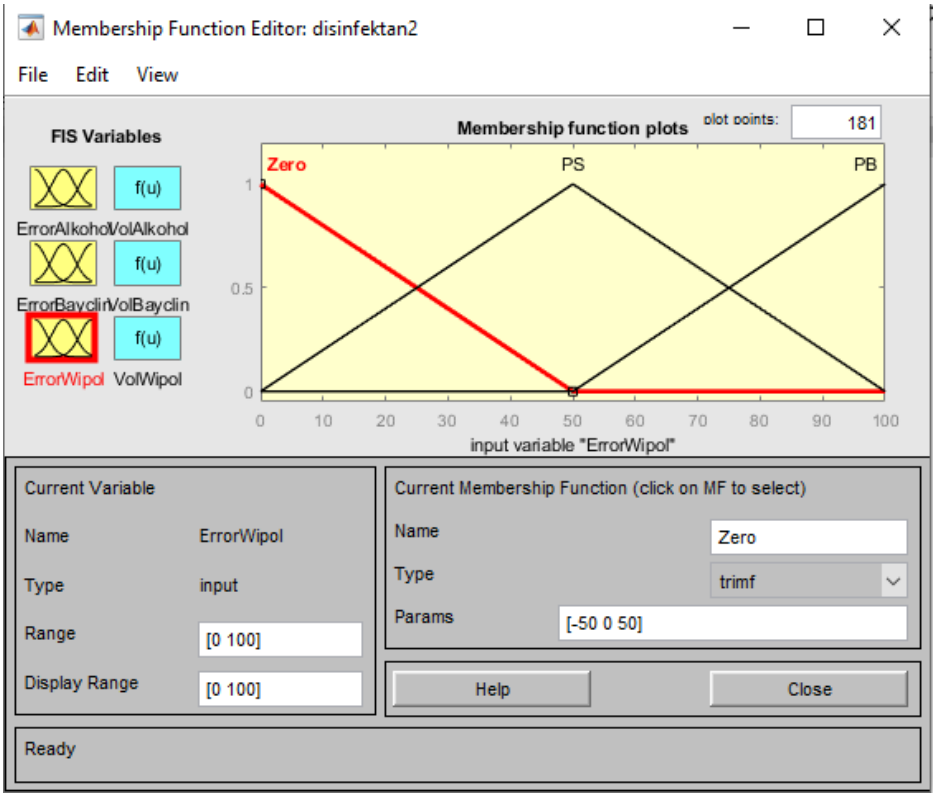
3.3.2 Perancangan *Fuzzy* pada proses disinfektan

Pada proses disinfektan menggunakan 3 bahan olahan awal sebelum menjadi hasil yang diinginkan oleh pengguna/*user* terdapat bahan – bahan Alkohol (60%), Bayclin (20%), dan Wipol (20%). Proses *Fuzzy Logic* menggunakan metode Sugeno pada Tugas Akhir ini, terdapat 3 *input* dan 3 *output* yang sudah disesuaikan dengan peneliti dan didapatkan hasil akhir untuk *rule base*nya sebanyak 27 *rules* pada kedua proses akhir yaitu cairan sanitasi dan disinfektan. Gambar 3.14 menunjukkan *Fuzzy Logic Designer* untuk tabung sanitasi.



Gambar 3.14. *Fuzzy logic designer* disinfektan

Pada *input error* ketiga bahan disinfektan terdapat 3 fungsi keanggotaan yaitu *Zero, PS (*Positif *Small),* dan PB (Positif *Big*). Pada masing – masing *input*an memiliki *domain range* atau batasan nilai yang sesuai dengan data dan sudah dilakukan oleh peneliti. Untuk lebih jelasnya bisa dilihat pada Gambar 3.15 yang dimana menunjukkan kurva *domain range* atau batasan nilai sebagai contoh pada *input error* wipol.



Gambar 3.15. *Membership function input error* wipol

Tabel 3.2 merupakan batasan nilai untuk *input error* wipol.

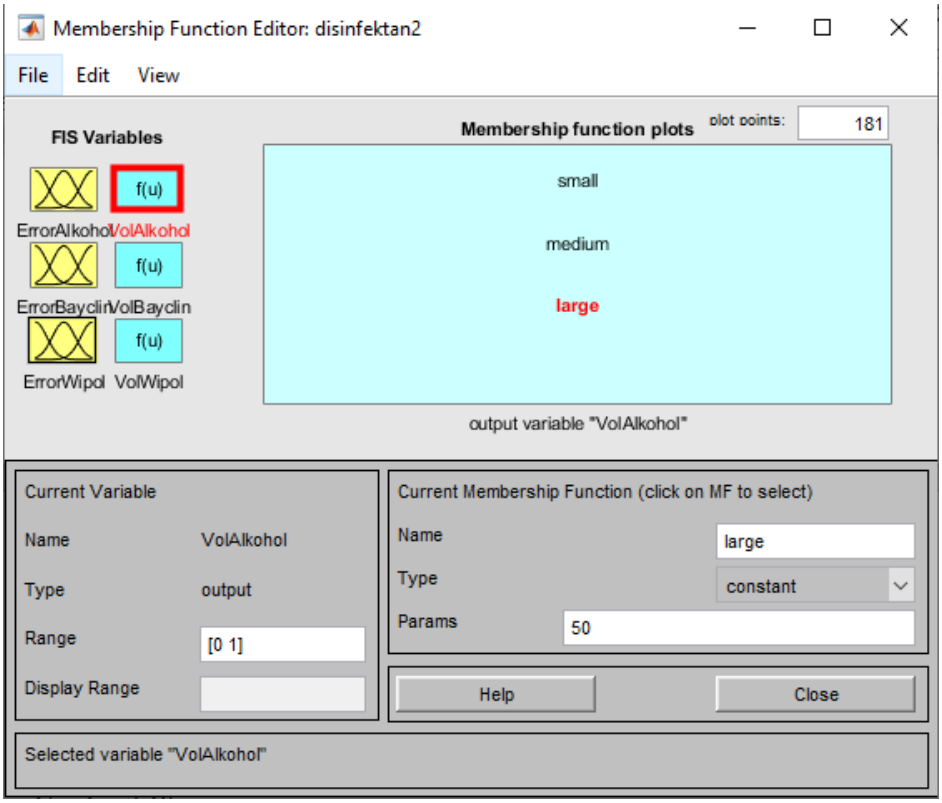
Tabel 3.2. Domain range input error Alkohol sanitasi

No.	Fungsi Keanggotaan	*Range* (%)
1	*Zero*	[0 ≤50]
2	Positif *Small* (PS)	[0 ≤100]
3	Positif *Big* (PB)	[50 ≥100]

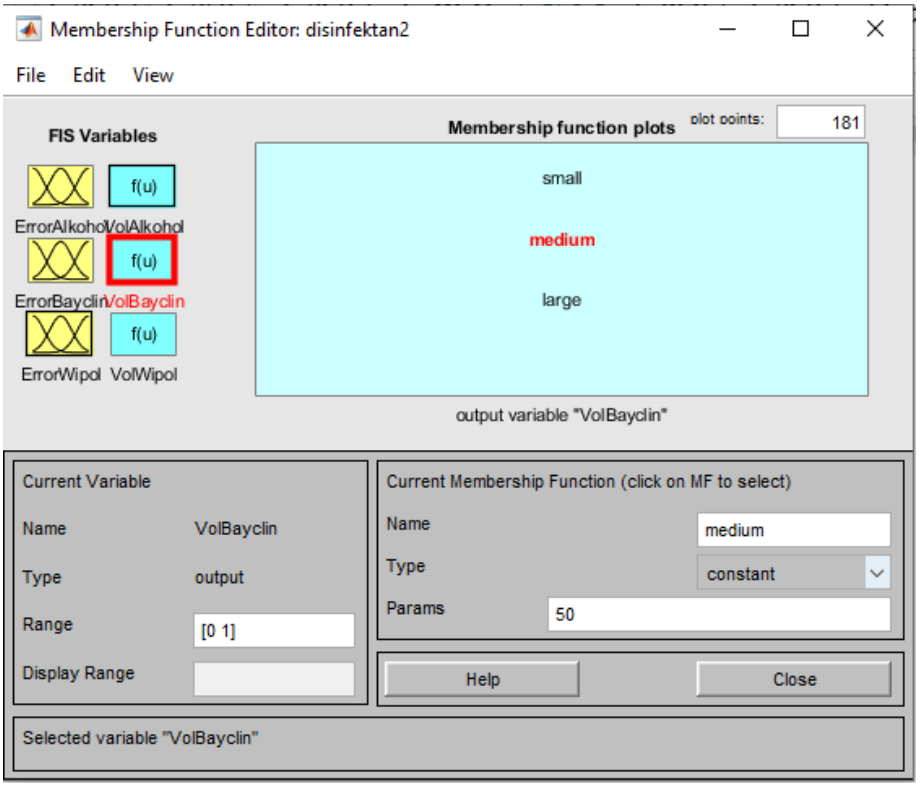
Untuk kedua bahan mengolah disinfektan yaitu alkohol dan bayclin sama seperti kurva *domain range* wipol beserta dengan isi tabel.

Pada perancangan kontroler ini selain memiliki 3 *input error*, juga memiliki 3 *output* yang terdiri dari 3 persentase volume, yaitu *volume* alkohol untuk disinfektan, *volume* bayclin, dan *volume* wipol. Berikut ini merupakan *membership function* dari ketiga set *output* yang dihasilkan yaitu *volume* alkohol untuk

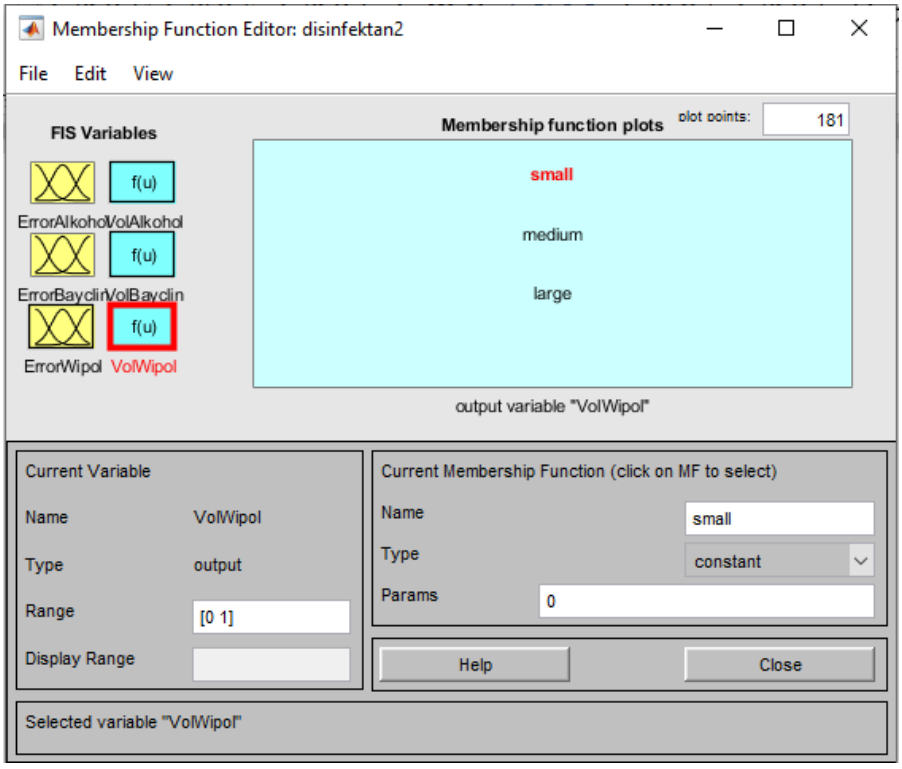
disinfektan yang dapat dilihat pada Gambar 3.16, *volume* bayclin yang dapat dilihat pada Gambar 3.17, dan *volume* wipol yang dapat dilihat pada Gambar 3.18.



Gambar 3.16. *Membership function output* alkohol untuk *range large*

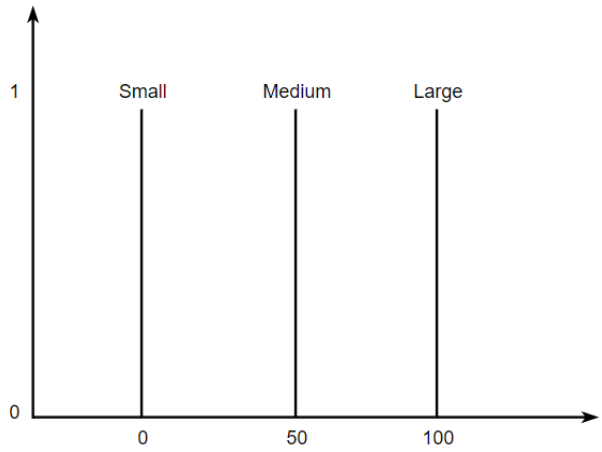


Gambar 3.17. *Membership function output* bahan bayclin untuk *range medium*



Gambar 3.18. *Membership function output* bahan wipol untuk *range small*

Pada keluaran *Fuzzy* dalam perancangan ini memiliki keluaran berupa *singleton* yang sama untuk *output* Bayclin, Wipol, dan Dettol dengan jangkauan *range* 0 sampai dengan 100, begitupula untuk hasil *output* alkohol disinfektan dengan jangkauan *range* 0 sampai dengan 50 menjadi 3 variabel lingustik. Variabel untuk *output* alkohol, bayclin, dan wipol. Small dengan nilai *singleton* 0, Medium dengan nilai *singleton* 50, Large dengan nilai *singleton* 100. Berikut pada Gambar 3.19 merupakan nilai *singleton output* Alkohol, Bayclin, dan Wipol yang digunakan.



Gambar 3.19. *Membership function output* alkohol, bayclin, dan wipol

Rule base yang diperoleh dari perancangan kontroler ini berjumlah 27 *rules* yang dapat dilihat sebagai berikut :

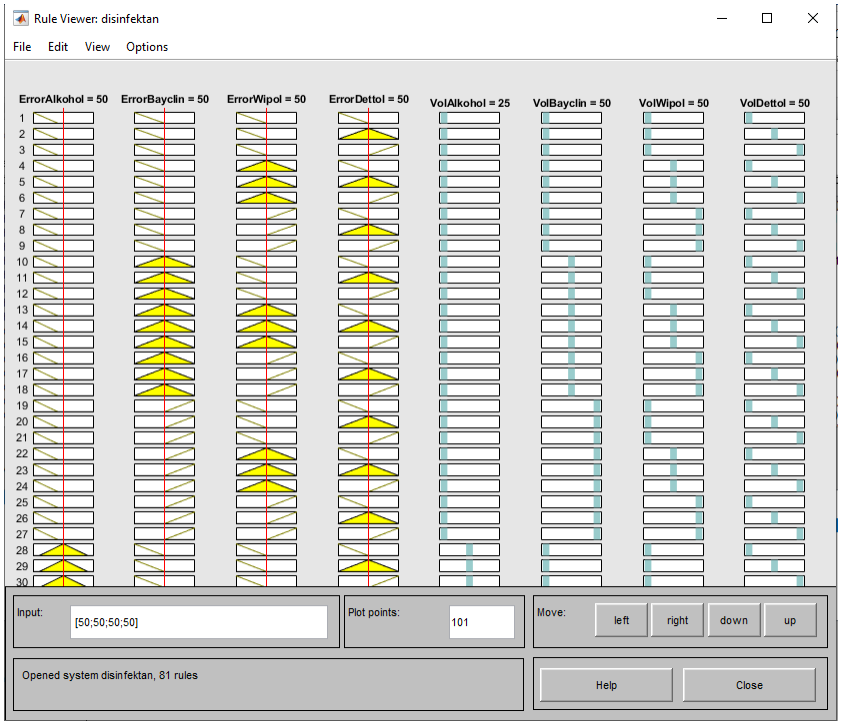
1. Jika *Error Alkohol* adalah *Zero* && *Error Bayclin* adalah *Zero* && *Error Wipol* adalah *Zero* maka Volume *Alkohol* adalah Small && Volume *Bayclin* adalah Small && Volume *Wipol* adalah Small.
2. Jika *Error Alkohol* adalah *Zero* && *Error Bayclin* adalah *Zero* && *Error Wipol* adalah PS maka Volume *Alkohol* adalah Small && Volume *Bayclin* adalah Small && Volume *Wipol* adalah Medium.
3. Jika *Error Alkohol* adalah *Zero* && *Error Bayclin* adalah *Zero* && *Error Wipol* adalah PB maka Volume *Alkohol* adalah Small && Volume *Bayclin* adalah Small && Volume *Wipol* adalah Large.

4. Jika *Error Alkohol* adalah *Zero* && *Error Bayclin* adalah PS && *Error Wipol* adalah *Zero* maka Volume *Alkohol* adalah Small && Volume *Bayclin* adalah Medium && Volume *Wipol* adalah Small.
5. Jika *Error Alkohol* adalah *Zero* && *Error Bayclin* adalah PS && *Error Wipol* adalah PS maka Volume *Alkohol* adalah Small && Volume *Bayclin* adalah Medium && Volume *Wipol* adalah Medium.
6. Jika *Error Alkohol* adalah *Zero* && *Error Bayclin* adalah PS && *Error Wipol* adalah PB maka Volume *Alkohol* adalah Small && Volume *Bayclin* adalah Medium && Volume *Wipol* adalah Large.
7. Jika *Error Alkohol* adalah *Zero* && *Error Bayclin* adalah PB && *Error Wipol* adalah *Zero* maka Volume *Alkohol* adalah Small && Volume *Bayclin* adalah Small && Volume *Wipol* adalah Large.
8. Jika *Error Alkohol* adalah *Zero* && *Error Bayclin* adalah PB && *Error Wipol* adalah PS maka Volume *Alkohol* adalah Small && Volume *Bayclin* adalah Medium && Volume *Wipol* adalah Large.
9. Jika *Error Alkohol* adalah *Zero* && *Error Bayclin* adalah PB && *Error Wipol* adalah PB maka Volume *Alkohol* adalah Small && Volume *Bayclin* adalah Large && Volume *Wipol* adalah Large.
10. Jika *Error Alkohol* adalah PS && *Error Bayclin* adalah *Zero* && *Error Wipol* adalah *Zero* maka Volume *Alkohol* adalah Medium && Volume *Bayclin* adalah Small && Volume *Wipol* adalah Small.
11. Jika *Error Alkohol* adalah PS && *Error Bayclin* adalah *Zero* && *Error Wipol* adalah PS maka Volume *Alkohol* adalah Medium && Volume *Bayclin* adalah Small && Volume *Wipol* adalah Medium.
12. Jika *Error Alkohol* adalah PS && *Error Bayclin* adalah *Zero* && *Error Wipol* adalah PB maka Volume *Alkohol* adalah Medium && Volume *Bayclin* adalah Small && Volume *Wipol* adalah Large.
13. Jika *Error Alkohol* adalah PS && *Error Bayclin* adalah PS && *Error Wipol* adalah *Zero* maka Volume *Alkohol* adalah Medium && Volume *Bayclin* adalah Medium && Volume *Wipol* adalah Small.

14. Jika *Error Alkohol* adalah PS && *Error Bayclin* adalah PS && *Error Wipol* adalah PS maka Volume *Alkohol* adalah Medium && Volume *Bayclin* adalah Medium && Volume *Wipol* adalah Medium.
15. Jika *Error Alkohol* adalah PS && *Error Bayclin* adalah PS && *Error Wipol* adalah PB maka Volume *Alkohol* adalah Medium && Volume *Bayclin* adalah Medium && Volume *Wipol* adalah Large.
16. Jika *Error Alkohol* adalah PS && *Error Bayclin* adalah PB && *Error Wipol* adalah *Zero* maka Volume *Alkohol* adalah Medium && Volume *Bayclin* adalah Large && Volume *Wipol* adalah Small.
17. Jika *Error Alkohol* adalah PS && *Error Bayclin* adalah PB && *Error Wipol* adalah PS maka Volume *Alkohol* adalah Medium && Volume *Bayclin* adalah Large && Volume *Wipol* adalah Medium.
18. Jika *Error Alkohol* adalah PS && *Error Bayclin* adalah PB && *Error Wipol* adalah PB maka Volume *Alkohol* adalah Medium && Volume *Bayclin* adalah Large && Volume *Wipol* adalah Large.
19. Jika *Error Alkohol* adalah PB && *Error Bayclin* adalah *Zero* && *Error Wipol* adalah *Zero* maka Volume *Alkohol* adalah Large && Volume *Bayclin* adalah Small && Volume *Wipol* adalah Small.
20. Jika *Error Alkohol* adalah PB && *Error Bayclin* adalah *Zero* && *Error Wipol* adalah PS maka Volume *Alkohol* adalah Large && Volume *Bayclin* adalah Small && Volume *Wipol* adalah Medium.
21. Jika *Error Alkohol* adalah PB && *Error Bayclin* adalah *Zero* && *Error Wipol* adalah PB maka Volume *Alkohol* adalah Large && Volume *Bayclin* adalah Small && Volume *Wipol* adalah Large.
22. Jika *Error Alkohol* adalah PB && *Error Bayclin* adalah PS && *Error Wipol* adalah *Zero* maka Volume *Alkohol* adalah Large && Volume *Bayclin* adalah Medium && Volume *Wipol* adalah Small.
23. Jika *Error Alkohol* adalah PB && *Error Bayclin* adalah PS && *Error Wipol* adalah PS maka Volume *Alkohol* adalah Large && Volume *Bayclin* adalah Medium && Volume *Wipol* adalah Medium.

24. Jika *Error Alkohol* adalah PB && *Error Bayclin* adalah PS && *Error Wipol* adalah PB maka Volume *Alkohol* adalah Large && Volume *Bayclin* adalah Medium && Volume *Wipol* adalah Large.
25. Jika *Error Alkohol* adalah PB && *Error Bayclin* adalah PB && *Error Wipol* adalah *Zero* maka Volume *Alkohol* adalah Large && Volume *Bayclin* adalah Large && Volume *Wipol* adalah Small.
26. Jika *Error Alkohol* adalah PB && *Error Bayclin* adalah PB && *Error Wipol* adalah PS maka Volume *Alkohol* adalah Large && Volume *Bayclin* adalah Large && Volume *Wipol* adalah Medium.
27. Jika *Error Alkohol* adalah PB && *Error Bayclin* adalah PB && *Error Wipol* adalah PB maka Volume *Alkohol* adalah Large && Volume *Bayclin* adalah Large && Volume *Wipol* adalah Large.

Pada Gambar 3.20 menunjukkan *rule viewer* sanitasi pada aplikasi simulasi MATLAB yang dapat disimulasikan menggunakan logika *Fuzzy* dengan tiga masukkan dan tiga keluaran nilai sebagai berikut.



Gambar 3.20. *Rule viewer* disinfektan

3.4 Perancangan Perangkat Lunak

Pada perancangan perangkat lunak terdiri dari penjelasan mengenai program inisialisasi dan konfigurasi pin, program pengendalian relai 12V untuk buka tutup *Pompa air DC 12V*, program pembacaan sensor *load cell* yang menyimpan ke EEPROM, dan program *Fuzzy* untuk mengontrol sistem sesuai dengan algoritma yang telah ditentukan. Bahasa yang digunakan merupakan Bahasa C yang deprogram menggunakan IDE Arduino.

3.4.1 Pemrograman Inisialisasi dan Konfigurasi Pin

Program pada keseluruhan sistem ini dirancang dengan mengikuti algoritma dan *flowchart* yang telah dibaca pada Mikrokontroler, yaitu Mikrokontroler ATMega 2560 Pro Mini. Berikut merupakan bagian inisialisasi dan konfigurasi pin pada Mikrokontoler ATMega 2560 Pro Mini.

```
#include <HX711_ADC.h>
#if defined(ESP8266)|| defined(ESP32) || defined(AVR)
#include <EEPROM.h>
#endif

//pins:
const int HX711_dout_1 = 6; //mcu > HX711 no 1 dout pin
const int HX711_sck_1 = 8;  //mcu > HX711 no 1 sck pin
const int HX711_dout_2 = 9; //mcu > HX711 no 2 dout pin
const int HX711_sck_2 = 10; //mcu > HX711 no 2 sck pin

//HX711 constructor (dout pin, sck pin)
HX711_ADC LoadCell_1(HX711_dout_1, HX711_sck_1); //HX711 1
HX711_ADC LoadCell_2(HX711_dout_2, HX711_sck_2); //HX711 2

const int calVal_eepromAdress_1 = 0; // eeprom adress for
calibration value load cell 1 (4 bytes)
const int calVal_eepromAdress_2 = 4; // eeprom adress for
calibration value load cell 2 (4 bytes)
unsigned long t = 0;
```

```
// pins declare for sanitasi
#define relay7 15 //digital pin relay bahan Alkohol Sanitasi
#define relay1 23 //digital pin relay bahan Wipol
#define relay4 5  //digital pin relay bahan H2O2

// pins declare for disinfektan
#define relay2 3  //digital pin relay bahan Alkohol DE
#define relay3 4  //digital pin relay bahan Bayclin
#define relay6 14 //digital pin relay bahan Wipol
#define relayM2 7 //digital pin relay bahan Dettol

// pins declare for buffer
#define relay8 16 //digital pin relay Buffer Sanitasi
#define relay9 17 //digital pin relay Buffer Desinfektan

// pin for LoadRelay
#define LoadRelay 27
```

3.4.2 Pemrograman untuk Pompa air DC 12V

Berikut adalah senarai algoritma kontrol ON dan OFF untuk sistem Pompa air DC 12V pada sistem alat dispenser otomatis dengan menggunakan relai pada setiap pompa yang diberikan sesuai dengan yang diuji oleh peneliti.

Baris program pada konfigurasi pin yang digunakan dalam bahasa C pemrograman IDE Arduino adalah sebagai berikut :

```
// pins declare for sanitasi
#define relay7 15 //digital pin relay bahan Alkohol Sanitasi
#define relay1 23 //digital pin relay bahan Wipol
#define relay4 5  //digital pin relay bahan H2O2

// pins declare for disinfektan
#define relay2 3  //digital pin relay bahan Alkohol DE
#define relay3 4  //digital pin relay bahan Bayclin
#define relay6 14 //digital pin relay bahan Wipol

// pins declare for buffer
```

```
#define relay8 16 //digital pin relay Buffer Sanitasi
#define relay9 17 //digital pin relay Buffer Desinfektan

// pin for LoadRelay
#define LoadRelay 27
```

Pada bagian berikut adalah senarai pemrograman untuk menginisiasi relai dimana bekerja sebagai *output* dalam *void setup* begitupula untuk *trigger* relai apakah *high trigger* atau *low trigger*, sebagai berikut :

```
void setup(){
  Serial.begin(9600);

  pinMode(relay1, OUTPUT);
  pinMode(relay2, OUTPUT);
  pinMode(relay3, OUTPUT);
  pinMode(relay4, OUTPUT);
  pinMode(relay6, OUTPUT);
  pinMode(relay7, OUTPUT);
  pinMode(relay8, OUTPUT);
  pinMode(relay9, OUTPUT);
  pinMode(LoadRelay, OUTPUT);

  digitalWrite(relay1, HIGH);
  digitalWrite(relay2, HIGH);
  digitalWrite(relay3, HIGH);
  digitalWrite(relay4, HIGH);
  digitalWrite(relay6, HIGH);
  digitalWrite(relay7, HIGH);
  digitalWrite(relay8, HIGH);
  digitalWrite(relay9, HIGH);
  digitalWrite(LoadRelay, LOW);}
```

Pada bagian ini adalah inisiasi fungsi – fungsi yang diawali dengan pembuatan sanitasi dari tabung bahan menuju tabung *buffer/mixer* sanitasi,

kemudian begitu pula dengan sistem pembuatan disinfektan dari tabung bahan menuju *buffer/mixer* disinfektan, sebagai berikut :

```
void HandSanitizer() {
  digitalWrite(relay1, LOW);
  digitalWrite(relay4, HIGH);
  digitalWrite(relay7, HIGH);
  delay(1000);
  digitalWrite(relay1, HIGH);
  digitalWrite(relay4, LOW);
  digitalWrite(relay7, HIGH);
  delay(1000);
  digitalWrite(relay1, HIGH);
  digitalWrite(relay4, HIGH);
  digitalWrite(relay7, LOW);
  delay(1000);
  digitalWrite(relay1, HIGH);
  digitalWrite(relay4, HIGH);
  digitalWrite(relay7, HIGH);
  delay(1000);
}

void Disinfektan() {
  digitalWrite(relay2, LOW);
  digitalWrite(relay3, HIGH);
  digitalWrite(relay6, HIGH);
  delay(1000);
  digitalWrite(relay2, HIGH);
  digitalWrite(relay3, LOW);
  digitalWrite(relay6, HIGH);
  delay(1000);
  digitalWrite(relay2, HIGH);
  digitalWrite(relay3, HIGH);
  digitalWrite(relay6, LOW);
  delay(1000);
  digitalWrite(relay2, HIGH);
  digitalWrite(relay3, HIGH);
```

```
  digitalWrite(relay6, HIGH);
  delay(1000);
}

void BufferHandSanitizer() {
  digitalWrite (relay8, LOW);
  digitalWrite (relay9, HIGH);
  delay(1000);
  digitalWrite (relay8, HIGH);
  digitalWrite (relay9, HIGH);
  delay(1000);
}

void BufferDisinfektan() {
  digitalWrite (relay8, HIGH);
  digitalWrite (relay9, LOW);
  delay(1000);
  digitalWrite (relay8, HIGH);
  digitalWrite (relay9, HIGH);
  delay(1000);
}

void loop(){
  HandSanitizer();
  Disinfektan();
  BufferHandSanitizer();
  BufferDisinfektan();
}
```

3.4.3 Pemrograman Fungsi Pembacaan Sensor *Load Cell*

Program pada sistem berat ini dirancang dengan mengikuti algoritma dan *flowchart* yang telah tertuang pada Mikrokontroler ATMega2560 Pro Mini. Fungsi pembacaan sensor berat atau *load cell* ini bertujuan sebagai *trigger* untuk tabung *buffer* atau *mixer* pada sanitasi dan disinfektan.

```
#include <HX711_ADC.h>
#if defined(ESP8266)|| defined(ESP32) || defined(AVR)
```

```
#include <EEPROM.h>
#endif

//pins:
const int HX711_dout_1 = 6; //mcu > HX711 no 1 dout pin
const int HX711_sck_1 = 8; //mcu > HX711 no 1 sck pin
const int HX711_dout_2 = 9; //mcu > HX711 no 2 dout pin
const int HX711_sck_2 = 10; //mcu > HX711 no 2 sck pin

//HX711 constructor (dout pin, sck pin)
HX711_ADC LoadCell_1(HX711_dout_1, HX711_sck_1); //HX711 1
HX711_ADC LoadCell_2(HX711_dout_2, HX711_sck_2); //HX711 2

const int calVal_eepromAdress_1 = 0; // eeprom adress for
calibration value load cell 1 (4 bytes)
const int calVal_eepromAdress_2 = 4; // eeprom adress for
calibration value load cell 2 (4 bytes)
unsigned long t = 0;

void setup(){
  Serial.begin(57600); delay(5000);
  Serial.println();
  Serial.println("Starting...");

  float calibrationValue_1; // calibration value load cell 1
  float calibrationValue_2; // calibration value load cell 2

  calibrationValue_1 = 696.0; // uncomment this if you want
to set this value in the sketch
  calibrationValue_2 = 733.0; // uncomment this if you want
to set this value in the sketch
#if defined(ESP8266) || defined(ESP32)
  //EEPROM.begin(512); // uncomment this if you use ESP8266
and want to fetch the value from eeprom
#endif
  EEPROM.get(calVal_eepromAdress_1, calibrationValue_1); //
uncomment this if you want to fetch the value from eeprom
```

```
  EEPROM.get(calVal_eepromAdress_2, calibrationValue_2); //
uncomment this if you want to fetch the value from eeprom

  LoadCell_1.begin();
  LoadCell_2.begin();
  //LoadCell_1.setReverseOutput();
  //LoadCell_2.setReverseOutput();
  unsigned long stabilizingtime = 2000; // tare preciscion
can be improved by adding a few seconds of stabilizing time
  boolean _tare = true; //set this to false if you don't want
tare to be performed in the next step
  byte loadcell_1_rdy = 0;
  byte loadcell_2_rdy = 0;
  while ((loadcell_1_rdy + loadcell_2_rdy) < 2) { //run
startup, stabilization and tare, both modules simultaniously
    if       (!loadcell_1_rdy)       loadcell_1_rdy       =
LoadCell_1.startMultiple(stabilizingtime, _tare);
    if       (!loadcell_2_rdy)       loadcell_2_rdy       =
LoadCell_2.startMultiple(stabilizingtime, _tare);
  }
  if (LoadCell_1.getTareTimeoutFlag()) {
    Serial.println("Timeout, check MCU>HX711 no.1 wiring and
pin designations");
  }
  if (LoadCell_2.getTareTimeoutFlag()) {
    Serial.println("Timeout, check MCU>HX711 no.2 wiring and
pin designations");
  }
  LoadCell_1.setCalFactor(calibrationValue_1); // user set
calibration value (float)
  LoadCell_2.setCalFactor(calibrationValue_2); // user set
calibration value (float)
  Serial.println("Startup is complete");
```

Dalam pemrosesan *void loop* disini dijelaskan dalam *coding*-an bahwa sensor *load cell.*

```
void loop() {
  static boolean newDataReady = 0;
  const int serialPrintInterval = 2000; //increase value to
slow down serial print activity

  // check for new data/start next conversion:
  if (LoadCell_1.update()) newDataReady = true;
  LoadCell_2.update();

  //get smoothed value from data set
  if ((newDataReady)) {
    if (millis() > t + serialPrintInterval) {
      float a = LoadCell_1.getData();
      float b = LoadCell_2.getData();
      Serial.print("Load_cell 1 output val: ");
      Serial.print(a);
      Serial.print("    Load_cell 2 output val: ");
      Serial.println(b);
      newDataReady = 0;
      t = millis();
}
  }

  //  receive  command  from  serial  terminal,  send  't'  to
initiate tare operation:
  if (Serial.available() > 0) {
    char inByte = Serial.read();
    if (inByte == 't') {
      LoadCell_1.tareNoDelay();
      LoadCell_2.tareNoDelay();
    }
  }

  //check if last tare operation is complete
  if (LoadCell_1.getTareStatus() == true) {
    Serial.println("Tare load cell 1 complete");
  }
  if (LoadCell_2.getTareStatus() == true) {
```

```
    Serial.println("Tare load cell 2 complete");
  }
}
```

3.4.4 Pemrograman Fungsi Pembacaan Batasan Sensor *Loadcell*

Fungsi pembacaan sensor berat atau *load cell* ini bertujuan sebagai *trigger* untuk tabung *buffer* atau *mixer* pada sanitasi dan disinfektan dimana mempunyai batasan minimum (≤ 200 gram) dan batasan maksimum (≤ 1250 gram).

```
float Trigger(){
      digitalWrite(LoadRelay, LOW);
      if (a <= 200){
        digitalWrite(relay1, LOW);
        digitalWrite(relay4, LOW);
        digitalWrite(relay7, LOW);
      }
      else if (a >= 1250){
        digitalWrite(relay1, HIGH);
        digitalWrite(relay4, HIGH);
        digitalWrite(relay7, HIGH);
    }
    if (b <= 200){
        digitalWrite(relay2, LOW);
        digitalWrite(relay3, LOW);
        digitalWrite(relay6, LOW);
    }
    else if (b >= 1250){
        digitalWrite(relay2, HIGH);
        digitalWrite(relay3, HIGH);
        digitalWrite(relay6, HIGH);
    }
    digitalWrite(LoadRelay, HIGH);
}

void loop (){
Trigger();
}
```

3.4.5 Pemrograman Algoritma Proses *Fuzzy* untuk Sanitasi

Algoritma *Fuzzy* yang digunakan ini berdasarkan atas rancangan *Fuzzy* dengan metode Sugeno yang telah peneliti buat. Memiliki 3 *input* sistem *Fuzzy* yaitu data *error* tabung bahan dan 3 *output* Volume tabung *mixer*/*buffer* sehingga diharapkan dapat menghasilkan presentase volume yang tepat untuk mendapatkan hasil pencampuran yang sesuai dengan *set point.* Selain itu, kontroler ini memiliki cara kerja yang dimana ketika data *error* masuk kedalam suatu *range* yaitu *Zero* maka besar presentase volume akan bernilai *small*, sedangkan jika data *error* masuk kedalam suatu *range* yaitu PS atau *Positive Small* maka besar presentase volume akan bernilai *medium*, dan jika data *error* masuk kedalam suatu *range* yaitu PB atau *Positive Big* maka presentase volume akan bernilai *large*. Berikut pada Tabel 3.3 merupakan kondisi data dimana *error* adalah untuk pengendalian volume.

Tabel 3.3. Kondisi data error untuk pengendalian volume

Data *Error* Bahan	**Interval *range***	**Volume Sanitasi**
Alkohol Sanitasi Zero	$0 \leq x \leq 50$	*Small*
Alkohol Sanitasi PS	$0 \leq x \leq 50$ $50 \leq x \leq 100$	*Medium*
Alkohol Sanitasi PB	$50 \leq x \leq 100$	*Large*
H_2O_2 Zero	$0 \leq x \leq 50$	*Small*
H_2O_2 PS	$0 \leq x \leq 50$ $50 \leq x \leq 100$	*Medium*
H_2O_2 PB	$50 \leq x \leq 100$	*Large*
Glycerin Zero	$0 \leq x \leq 50$	*Small*
Glycerin PS	$0 \leq x \leq 50$ $50 \leq x \leq 100$	*Medium*
Glycerin PB	$50 \leq x \leq 100$	*Large*

Berikut adalah senarai algoritma kontrol yang ada pada alat dispenser otomatis dengan *Fuzzy Logic Control* metode Sugeno Kontroler *Fuzzy* terdiri dari beberapa tahap. Langkah pertama adalah menginisialisasi variabel – variabel yang

digunakan pada kontroler beserta nilai awalnya. Lalu selanjutnya ada proses fuzzifikasi, *rule base*, dan defuzzifikasi.

Inisialisasi variabel untuk *Fuzzy Logic Control* terdiri dari inisialisasi *error* bahan, inisialisasi *membership function input Fuzzy*, inisialisasi *rule base*, dan inisialisasi *defuzzyfikasi*. Berikut adalah basis program proses inisialisasi yang telah dirancang :

```
/////////////// INISIALISASI VARIABEL FUZZY
float ErrorAlkohol;
float ErrorH2O2;
float ErrorGlycerin;
float Alkohol = 10,20;
float H2O2 = 12,61;
float Glycerin = 10,78;

/////////////// MEMBERSHIP FUNCTION INPUT FUZZY
float ErrorAlkoholZ, ErrorAlkoholPS, ErrorAlkoholPB,
ErrorH2O2Z, ErrorH2O2PS, ErrorH2O2PB,
ErrorGlycerinZ, ErrorGlycerinPS, ErrorGlycerinPB;

/////////////// RULE FUZZY
float rule1, rule1a, rule2, rule2a, rule3, rule3a, rule4,
rule4a, rule5, rule5a, rule6, rule6a, rule7, rule7a, rule8,
rule8a, rule9, rule9a, rule10, rule10a, rule11, rule11a,
rule12, rule12a, rule13, rule13a, rule14, rule14a, rule15,
rule15a, rule16, rule16a, rule17, rule17a, rule18, rule18a,
rule19, rule19a, rule20, rule20a, rule21, rule21a, rule22,
rule22a, rule23, rule23a, rule24, rule24a, rule25, rule25a,
rule26, rule26a, rule27, rule27a;

/////////////// OUTPUT DEFUZZYFIKASI
float VolumeAlkohol;
float VolumeH2O2;
float VolumeGlycerin;
```

Perancangan *software* pada *membership function* disesuaikan dengan perancangan kontroler *Fuzzy* yang telah dirancang pada subbab 3.4. yang terdiri dari 3 *membership function input* (*error alkohol*, *error* H_2O_2, dan *error Glycerin*). Bagian dari baris program proses fuzzifikasi yang telah dirancang adalah sebagai berikut :

```
/////////////// FUZZYFIKASI ERROR ALKOHOL
float FuzzyfikasiErrorAlkohol(){
float spAlkohol = 0;
ErrorAlkohol = spAlkohol + Alkohol;
Serial.print("Error Alkohol:"); delay(2000);
Serial.println(ErrorAlkohol);

//Alkohol Zero
if(ErrorAlkohol >= 0 && ErrorAlkohol < 50){
ErrorAlkoholZ = (50 - ErrorAlkohol)/50;
}
else if (ErrorAlkohol > 50 ) {
ErrorAlkoholZ= 0;
}
//Alkohol PS
if (ErrorAlkohol <= 0){
ErrorAlkoholPS= 0;
}
else if (ErrorAlkohol > 0 && ErrorAlkohol <= 50){
ErrorAlkoholPS= (ErrorAlkohol-0)/50;
}
else if (ErrorAlkohol > 50 && ErrorAlkohol < 100){
ErrorAlkoholPS= (100-ErrorAlkohol)/50;
}
else if (ErrorAlkohol >= 100){
ErrorAlkoholPS= 0;
}

//Alkohol PB
if (ErrorAlkohol <= 50){
```

```
ErrorAlkoholPB= 0;
}
else if (ErrorAlkohol > 50 && ErrorAlkohol <= 100){
ErrorAlkoholPB= (ErrorAlkohol-50)/50;
}
else if (ErrorAlkohol > 100){
ErrorAlkoholPB= 1;
}
}

/////////////// FUZZYFIKASI ERROR H2O2
float FuzzyfikasiErrorH2O2() {
float spH2O2 = 0;
ErrorH2O2 = spH2O2 + H2O2;
Serial.print("Error H2O2:"); delay(2000);
Serial.println(ErrorH2O2);

//H2O2 Zero
if (ErrorH2O2 >= 0 && ErrorH2O2 < 50){
  ErrorH2O2Z= (50 - ErrorH2O2)/50;
}
else if (ErrorH2O2 >= 50){
ErrorH2O2Z= 0;
}

//H2O2 PS
if (ErrorH2O2 <= 0){
ErrorH2O2PS= 0;
}
else if (ErrorH2O2 > 0 && ErrorH2O2 <= 50){
ErrorH2O2PS= (ErrorH2O2-0)/50;
}
else if (ErrorH2O2 > 50 && ErrorH2O2 < 100){
ErrorH2O2PS= (100-ErrorH2O2)/50;
}
else if (ErrorH2O2 >= 100){
ErrorH2O2PS= 0;
}
```

```
//H2O2 PB
if(ErrorH2O2 <= 50){
ErrorH2O2PB= 0;
}
else if (ErrorH2O2 > 50 && ErrorH2O2 <= 100){
ErrorH2O2PB= (ErrorH2O2 - 50)/50;
}
else if (ErrorH2O2 > 100){
ErrorH2O2PB= 1;
}
//Serial.print("Error H2O2 Zero:"); delay(2000);
}

/////////////// FUZZYFIKASI ERROR GLYCERIN
float FuzzyfikasiErrorGlycerin() {
float spGlycerin = 0;
ErrorGlycerin = spGlycerin + Glycerin;
Serial.print("Error Glycerin:"); delay(2000);
Serial.println(ErrorGlycerin);

//Glycerin Zero
if(ErrorGlycerin >= 0 && ErrorGlycerin < 50){
ErrorGlycerinZ= (50 - ErrorGlycerin)/50;
}
else if (ErrorGlycerin >= 50){
ErrorGlycerinZ= 0;
}

//Glycerin PS
if (ErrorGlycerin <= 0){
ErrorGlycerinPS= 0;
}
else if (ErrorGlycerin > 0 && ErrorGlycerin <= 50){
ErrorGlycerinPS= (ErrorGlycerin-0)/50;
}
else if (ErrorGlycerin > 50 && ErrorGlycerin <= 100){
ErrorGlycerinPS= (100-ErrorGlycerin)/50;
```

```
}
else if (ErrorGlycerin > 100){
ErrorGlycerinPS= 0;
}

//Glycerin PB
if (ErrorGlycerin <= 50){
ErrorGlycerinPB= 0;
}
else if (ErrorGlycerin > 50 && ErrorGlycerin <= 100){
ErrorGlycerinPB= (ErrorGlycerin - 50)/50;
}
else if (ErrorGlycerin > 100){
ErrorGlycerinPB= 1;
}
}
```

Perancangan *software* untuk menentukan *rule base* terdiri dari 27 set aturan yang menggunakan aturan *min* dimana nilai minimum dari hasil perhitungan yang akan diambil sesuai pada Gambar 3.13. Sehingga dalam bahasa pemrograman dapat dituliskan sebagai berikut :

```
/////////////// RULE BASE
float Rule(){
rule1 = min (ErrorAlkoholZ, ErrorH2O2Z);
rule1a = min (rule1, ErrorGlycerinZ);

rule2 = min (ErrorAlkoholZ, ErrorH2O2Z);
rule2a = min (rule2, ErrorGlycerinPS);

rule3 = min (ErrorAlkoholZ, ErrorH2O2Z);
rule3a = min (rule3, ErrorGlycerinPB);

rule4 = min (ErrorAlkoholZ, ErrorH2O2PS);
rule4a = min (rule4, ErrorGlycerinZ);

rule5 = min (ErrorAlkoholZ, ErrorH2O2PS);
```

```
rule5a = min (rule5, ErrorGlycerinPS);

rule6 = min (ErrorAlkoholZ, ErrorH2O2PS);
rule6a = min (rule6, ErrorGlycerinPB);

rule7 = min (ErrorAlkoholZ, ErrorH2O2PB);
rule7a = min (rule7, ErrorGlycerinZ);

rule8 = min (ErrorAlkoholZ, ErrorH2O2PB);
rule8a = min (rule8, ErrorGlycerinPS);

rule9 = min (ErrorAlkoholZ, ErrorH2O2PB);
rule9a = min (rule9, ErrorGlycerinPB);

rule10 = min (ErrorAlkoholPS, ErrorH2O2Z);
rule10a = min (rule10, ErrorGlycerinZ);

rule11 = min (ErrorAlkoholPS, ErrorH2O2Z);
rule11a = min (rule11, ErrorGlycerinPS);

rule12 = min (ErrorAlkoholPS, ErrorH2O2Z);
rule12a = min (rule12, ErrorGlycerinPB);

rule13 = min (ErrorAlkoholPS, ErrorH2O2PS);
rule13a = min (rule13, ErrorGlycerinZ);

rule14 = min (ErrorAlkoholPS, ErrorH2O2PS);
rule14a = min (rule14, ErrorGlycerinPS);

rule15 = min (ErrorAlkoholPS, ErrorH2O2PS);
rule15a = min (rule15, ErrorGlycerinPB);

rule16 = min (ErrorAlkoholPS, ErrorH2O2PB);
rule16a = min (rule16, ErrorGlycerinZ);

rule17 = min (ErrorAlkoholPS, ErrorH2O2PB);
rule17a = min (rule17, ErrorGlycerinPS);
```

```
rule18 = min (ErrorAlkoholPS, ErrorH2O2PB);
rule18a = min (rule18, ErrorGlycerinPB);

rule19 = min (ErrorAlkoholPB, ErrorH2O2Z);
rule19a = min (rule19, ErrorGlycerinZ);

rule20 = min (ErrorAlkoholPB, ErrorH2O2Z);
rule20a = min (rule20, ErrorGlycerinPS);

rule21 = min (ErrorAlkoholPB, ErrorH2O2Z);
rule21a = min (rule21, ErrorGlycerinPB);

rule22 = min (ErrorAlkoholPB, ErrorH2O2PS);
rule22a = min (rule22, ErrorGlycerinZ);

rule23 = min (ErrorAlkoholPB, ErrorH2O2PS);
rule23a = min (rule23, ErrorGlycerinPS);

rule24 = min (ErrorAlkoholPB, ErrorH2O2PS);
rule24a = min (rule24, ErrorGlycerinPB);

rule25 = min (ErrorAlkoholPB, ErrorH2O2PB);
rule25a = min (rule25, ErrorGlycerinZ);

rule26 = min (ErrorAlkoholPB, ErrorH2O2PB);
rule26a = min (rule26, ErrorGlycerinPS);

rule27 = min (ErrorAlkoholPB, ErrorH2O2PB);
rule27a = min (rule27, ErrorGlycerinPB);
}
```

Proses *defuzzyfikasi* merupakan tahap akhir dalam pengolahan kontroler *Fuzzy Logic Controler* yang digunakan dalam penelitian kali ini yaitu metode Sugeno. Proses ini menggunakan prinsip *Weight of Average* (WOA) dengan rumus membagi jumlah dari perkalian antara *rule* dengan nilai keluaran dengan jumlah *rule.* Penerapan dalam proses *defuzzyfikasi* bisa dilihat sebagai berikut untuk proses sanitasi :

```
float Defuzzyfikasi() {
//OUTPUT Volume Alkohol (dalam satuan %)
float BKA = 0;
float BSA = 25;
float BPA = 50;
//OUTPUT Volume H2O2 (dalam satuan %)
float BKH = 0;
float BSH = 50;
float BPH = 100;
//Output Volume Glycerin(dalam satuan %)
float BKG = 0;
float BSG = 50;
float BPG = 100;

VolumeAlkohol = (rule1a*BKA + rule2a*BKA + rule3a*BKA +
rule4a*BKA + rule5a*BKA + rule6a*BKA + rule7a*BKA +
rule8a*BKA + rule9a*BKA + rule10a*BSA + rule11a*BSA +
rule12a*BSA + rule13a*BSA + rule14a*BSA + rule15a*BSA +
rule16a*BSA + rule17a*BSA + rule18a*BSA + rule19a*BPA +
rule20a*BPA + rule21a*BPA + rule22a*BPA + rule23a*BPA +
rule24a*BPA + rule25a*BPA + rule26a*BPA + rule27a*BPA)/
(rule1a + rule2a + rule3a + rule4a + rule5a + rule6a +
rule7a + rule8a + rule9a + rule10a + rule11a + rule12a +
rule13a + rule14a + rule15a + rule16a + rule17a + rule18a +
rule19a + rule20a + rule21a + rule22a + rule23a + rule24a +
rule25a + rule26a + rule27a);

VolumeH2O2 = (rule1a*BKH + rule2a*BKH + rule3a*BKH +
rule4a*BSH + rule5a*BSH + rule6a*BSH + rule7a*BPH +
rule8a*BPH + rule9a*BPH + rule10a*BKH + rule11a*BKH +
rule12a*BKH + rule13a*BSH + rule14a*BSH + rule15a*BSH +
rule16a*BPH + rule17a*BPH + rule18a*BPH + rule19a*BKH +
rule20a*BKH + rule21a*BKH + rule22a*BSH + rule23a*BSH +
rule24a*BSH     +     rule25a*BPH     +     rule26a*BPH     +
rule27a*BPH)/(rule1a + rule2a + rule3a + rule4a + rule5a +
rule6a + rule7a + rule8a + rule9a + rule10a + rule11a + rule12a
```

```
+ rule13a + rule14a + rule15a + rule16a + rule17a + rule18a
+ rule19a + rule20a + rule21a + rule22a + rule23a + rule24a
+ rule25a + rule26a + rule27a);

VolumeGlycerin = (rule1a*BKG + rule2a*BSG + rule3a*BPG +
rule4a*BKG + rule5a*BSG + rule6a*BPG + rule7a*BKG +
rule8a*BSG + rule9a*BPG + rule10a*BKG + rule11a*BSG +
rule12a*BPG + rule13a*BKG + rule14a*BSG + rule15a*BPG +
rule16a*BKG + rule17a*BSG + rule18a*BPG + rule19a*BKG +
rule20a*BSG + rule21a*BPG + rule22a*BKG + rule23a*BSG +
rule24a*BPG + rule25a*BKG + rule26a*BSG + rule27a*BPG)/(
rule1a + rule2a + rule3a + rule4a + rule5a + rule6a + rule7a
+ rule8a + rule9a + rule10a + rule11a + rule12a + rule13a +
rule14a + rule15a + rule16a + rule17a + rule18a + rule19a +
rule20a + rule21a + rule22a + rule23a + rule24a + rule25a +
rule26a + rule27a);
}

void setup() {
Serial.begin(57600);
Serial.print ("Starting Fuzzy Sanitasi...");
}
void loop() {
  FuzzyfikasiErrorAlkohol();
  FuzzyfikasiErrorH2O2();
  FuzzyfikasiErrorGlycerin();
  Rule();
  Defuzzyfikasi();
}
```

3.4.6 Pemrograman Inisialisasi Proses *Fuzzy* untuk Disinfektan

Algoritma *Fuzzy* yang digunakan ini berdasarkan atas rancangan *Fuzzy* dengan metode Sugeno yang telah peneliti buat. Memiliki 3 *input* sistem *Fuzzy* yaitu data *error* tabung bahan dan 3 *output* Volume tabung *mixer*/*buffer* sehingga diharapkan dapat menghasilkan presentase volume yang tepat untuk mendapatkan hasil pencampuran yang sesuai dengan *set point.* Selain itu, kontroler ini memiliki

cara kerja yang dimana ketika data *error* masuk kedalam suatu *range* yaitu *Zero* maka besar presentase volume akan bernilai *small*, sedangkan jika data *error* masuk kedalam suatu *range* yaitu PS atau *Positive Small* maka besar presentase volume akan bernilai *medium*, dan jika data *error* masuk kedalam suatu *range* yaitu PB atau *Positive Big* maka presentase volume akan bernilai *large*. Berikut pada Tabel 3.4 merupakan kondisi data dimana *error* adalah untuk pengendalian volume.

Tabel 3.4. Kondisi data error untuk pengendalian volume

Data *Error* Bahan	**Interval *range***	**Volume Sanitasi**
Alkohol Disinfektan Zero	$0 \leq x \leq 50$	*Small*
Alkohol Disinfektan PS	$0 \leq x \leq 50$ $50 \leq x \leq 100$	*Medium*
Alkohol Disinfektan PB	$50 \leq x \leq 100$	*Large*
Bayclin Zero	$0 \leq x \leq 50$	*Small*
Bayclin PS	$0 \leq x \leq 50$ $50 \leq x \leq 100$	*Medium*
Bayclin PB	$50 \leq x \leq 100$	*Large*
Wipol Zero	$0 \leq x \leq 50$	*Small*
Wipol PS	$0 \leq x \leq 50$ $50 \leq x \leq 100$	*Medium*
Wipol PB	$50 \leq x \leq 100$	*Large*

Berikut adalah senarai algoritma kontrol yang ada pada alat dispenser otomatis dengan *Fuzzy Logic Control* metode Sugeno Kontroler *Fuzzy* terdiri dari beberapa tahap. Langkah pertama adalah menginisialisasi variabel – variabel yang digunakan pada kontroler beserta nilai awalnya. Lalu selanjutnya ada proses fuzzifikasi, *rule base*, dan defuzzifikasi.

Inisialisasi variabel untuk *Fuzzy Logic Control* terdiri dari inisialisasi *error* bahan, inisialisasi *membership function input Fuzzy*, inisialisasi *rule base*, dan inisialisasi *defuzzyfikasi*. Berikut adalah basis program proses inisialisasi yang telah dirancang :

```
/////////////// INISIALISASI VARIABEL FUZZY
float ErrorAlkohol_1;
float ErrorBayclin;
float ErrorWipol;
float Alkohol_1 = 12,56;
float Bayclin = 15,58;
float Wipol = 10,61;

/////////////// MEMBERSHIP FUNCTION INPUT FUZZY
float ErrorAlkohol_1Z, ErrorAlkohol_1PS, ErrorAlkohol_1PB,
      ErrorBayclinZ, ErrorBayclinPS, ErrorBayclinPB,
      ErrorWipolZ, ErrorWipolPS, ErrorWipolPB;

/////////////// RULE FUZZY
float  rule1_1,  rule1a_1,  rule2_1,  rule2a_1,  rule3_1,
rule3a_1,  rule4_1,  rule4a_1,  rule5_1,  rule5a_1,  rule6_1,
rule6a_1,  rule7_1,  rule7a_1,  rule8_1,  rule8a_1,  rule9_1,
rule9a_1, rule10_1, rule10a_1, rule11_1, rule11a_1, rule12_1,
rule12a_1,   rule13_1,   rule13a_1,   rule14_1,   rule14a_1,
rule15_1,   rule15a_1,   rule16_1,   rule16a_1,   rule17_1,
rule17a_1,   rule18_1,   rule18a_1,   rule19_1,   rule19a_1,
rule20_1,   rule20a_1,   rule21_1,   rule21a_1,   rule22_1,
rule22a_1,   rule23_1,   rule23a_1,   rule24_1,   rule24a_1,
rule25_1,   rule25a_1,   rule26_1,   rule26a_1,   rule27_1,
rule27a_1;

/////////////// OUTPUT DEFUZZYFIKASI
float VolumeAlkohol_1;
float VolumeBayclin;
float VolumeWipol;
```

Perancangan *software* pada *membership function* disesuaikan dengan perancangan kontroler *Fuzzy* yang telah dirancang pada subbab 3.4. yang terdiri dari 3 *membership function input* (*error alkohol*, *error Bayclin*, dan *error Wipol*). Bagian dari baris program proses fuzzifikasi yang telah dirancang adalah sebagai berikut :

```
/////////////// FUZZYFIKASI ERROR ALKOHOL
void FuzzyfikasiErrorAlkohol_1(){
float spAlkohol_1 = 0;
ErrorAlkohol_1 = spAlkohol_1 + Alkohol_1;
Serial.print("Error Alkohol_1:");
Serial.println(ErrorAlkohol_1);

//Alkohol Zero
if(ErrorAlkohol_1 >= 0 && ErrorAlkohol_1 < 50){
ErrorAlkohol_1Z = (50 - ErrorAlkohol_1)/50;
}
else if (ErrorAlkohol_1 > 50 ) {
ErrorAlkohol_1Z= 0;
}
//Alkohol PS
if (ErrorAlkohol_1 <= 0){
ErrorAlkohol_1PS= 0;
}
else if (ErrorAlkohol_1 > 0 && ErrorAlkohol_1 <= 50){
ErrorAlkohol_1PS= (ErrorAlkohol_1-0)/50;
}
else if (ErrorAlkohol_1 > 50 && ErrorAlkohol_1 < 100){
ErrorAlkohol_1PS= (100-ErrorAlkohol_1)/50;
}
else if (ErrorAlkohol_1 >= 100){
ErrorAlkohol_1PS= 0;
}

//Alkohol PB
if (ErrorAlkohol_1 <= 50){
ErrorAlkohol_1PB= 0;
}
else if (ErrorAlkohol_1 > 50 && ErrorAlkohol_1 <= 100){
ErrorAlkohol_1PB= (ErrorAlkohol_1-50)/50;
}
else if (ErrorAlkohol_1 > 100){
ErrorAlkohol_1PB= 1;
}
```

```
}

/////////////// FUZZYFIKASI ERROR BAYCLIN
void FuzzyfikasiErrorBayclin() {
float spBayclin = 0;
ErrorBayclin = spBayclin + Bayclin;
Serial.print("Error Bayclin:");
Serial.println(ErrorBayclin);

//Bayclin Zero
if(ErrorBayclin >= 0 && ErrorBayclin < 50){
ErrorBayclinZ= (50 - ErrorBayclin)/50;
}
else if (ErrorBayclin >= 50){
ErrorBayclinZ= 0;
}

//Bayclin PS
if (ErrorBayclin <= 0){
ErrorBayclinPS= 0;
}
else if (ErrorBayclin > 0 && ErrorBayclin <= 50){
ErrorBayclinPS= (ErrorBayclin-0)/50;
}
else if (ErrorBayclin > 50 && ErrorBayclin <= 100){
ErrorBayclinPS= (100-ErrorBayclin)/50;
}
else if (ErrorBayclin > 100){
ErrorBayclinPS= 0;
}

//Bayclin PB
if (ErrorBayclin <= 50){
ErrorBayclinPB= 0;
}
else if (ErrorBayclin > 50 && ErrorBayclin <= 100){
ErrorBayclinPB= (ErrorBayclin - 50)/50;
}
```

```
else if (ErrorBayclin > 100){
ErrorBayclinPB= 1;
}
}

/////////////// FUZZYFIKASI ERROR Wipol
void FuzzyfikasiErrorWipol() {
float spWipol = 0;
ErrorWipol = spWipol + Wipol;
Serial.print("Error Wipol:");
Serial.println(ErrorWipol);

//Wipol Zero
if (ErrorWipol >= 0 && ErrorWipol < 50){
  ErrorWipolZ= (50 - ErrorWipol)/50;
}
else if (ErrorWipol >= 50){
ErrorWipolZ= 0;
}

//Wipol PS
if (ErrorWipol <= 0){
ErrorWipolPS= 0;
}
else if (ErrorWipol > 0 && ErrorWipol <= 50){
ErrorWipolPS= (ErrorWipol-0)/50;
}
else if (ErrorWipol > 50 && ErrorWipol < 100){
ErrorWipolPS= (100-ErrorWipol)/50;
}
else if (ErrorWipol >= 100){
ErrorWipolPS= 0;
}

//Wipol PB
if(ErrorWipol <= 50){
ErrorWipolPB= 0;
}
```

```
else if (ErrorWipol > 50 && ErrorWipol <= 100){
ErrorWipolPB= (ErrorWipol - 50)/50;
}
else if (ErrorWipol > 100){
ErrorWipolPB= 1;
}
Serial.print("Error Wipol Zero:");
}
```

Perancangan *software* untuk menentukan *rule base* terdiri dari 27 set aturan yang menggunakan aturan *min* dimana nilai minimum dari hasil perhitungan yang akan diambil sesuai pada Gambar 3.20. Sehingga dalam bahasa pemrograman dapat dituliskan sebagai berikut :

```
/////////////// RULE BASE
void Rule(){
rule1_1 = min (ErrorAlkohol_1Z, ErrorBayclinZ);
rule1a_1 = min (rule1_1, ErrorWipolZ);

rule2_1 = min (ErrorAlkohol_1Z, ErrorBayclinZ);
rule2a_1 = min (rule2_1, ErrorWipolPS);

rule3_1 = min (ErrorAlkohol_1Z, ErrorBayclinZ);
rule3a_1 = min (rule3_1, ErrorWipolPB);

rule4_1 = min (ErrorAlkohol_1Z, ErrorBayclinPS);
rule4a_1 = min (rule4_1, ErrorWipolZ);

rule5_1 = min (ErrorAlkohol_1Z, ErrorBayclinPS);
rule5a_1 = min (rule5_1, ErrorWipolPS);

rule6_1 = min (ErrorAlkohol_1Z, ErrorBayclinPS);
rule6a_1 = min (rule6_1, ErrorWipolPB);

rule7_1 = min (ErrorAlkohol_1Z, ErrorBayclinPB);
rule7a_1 = min (rule7_1, ErrorWipolZ);
```

```
rule8_1 = min (ErrorAlkohol_1Z, ErrorBayclinPB);
rule8a_1 = min (rule8_1, ErrorWipolPS);

rule9_1 = min (ErrorAlkohol_1Z, ErrorBayclinPB);
rule9a_1 = min (rule9_1, ErrorWipolPB);

rule10_1 = min (ErrorAlkohol_1PS, ErrorBayclinZ);
rule10a_1 = min (rule10_1, ErrorWipolZ);

rule11_1 = min (ErrorAlkohol_1PS, ErrorBayclinZ);
rule11a_1 = min (rule11_1, ErrorWipolPS);

rule12_1 = min (ErrorAlkohol_1PS, ErrorBayclinZ);
rule12a_1 = min (rule12_1, ErrorWipolPB);

rule13_1 = min (ErrorAlkohol_1PS, ErrorBayclinPS);
rule13a_1 = min (rule13_1, ErrorWipolZ);

rule14_1 = min (ErrorAlkohol_1PS, ErrorBayclinPS);
rule14a_1 = min (rule14_1, ErrorWipolPS);

rule15_1 = min (ErrorAlkohol_1PS, ErrorBayclinPS);
rule15a_1 = min (rule15_1, ErrorWipolPB);

rule16_1 = min (ErrorAlkohol_1PS, ErrorBayclinPB);
rule16a_1 = min (rule16_1, ErrorWipolZ);

rule17_1 = min (ErrorAlkohol_1PS, ErrorBayclinPB);
rule17a_1 = min (rule17_1, ErrorWipolPS);

rule18_1 = min (ErrorAlkohol_1PS, ErrorBayclinPB);
rule18a_1 = min (rule18_1, ErrorWipolPB);

rule19_1 = min (ErrorAlkohol_1PB, ErrorBayclinZ);
rule19a_1 = min (rule19_1, ErrorWipolZ);

rule20_1 = min (ErrorAlkohol_1PB, ErrorBayclinZ);
rule20a_1 = min (rule20_1, ErrorWipolPS);
```

```
        rule21_1 = min (ErrorAlkohol_1PB, ErrorBayclinZ);
        rule21a_1 = min (rule21_1, ErrorWipolPB);

        rule22_1 = min (ErrorAlkohol_1PB, ErrorBayclinPS);
        rule22a_1 = min (rule22_1, ErrorWipolZ);

        rule23_1 = min (ErrorAlkohol_1PB, ErrorBayclinPS);
        rule23a_1 = min (rule23_1, ErrorWipolPS);

        rule24_1 = min (ErrorAlkohol_1PB, ErrorBayclinPS);
        rule24a_1 = min (rule24_1, ErrorWipolPB);

        rule25_1 = min (ErrorAlkohol_1PB, ErrorBayclinPB);
        rule25a_1 = min (rule25_1, ErrorWipolZ);

        rule26_1 = min (ErrorAlkohol_1PB, ErrorBayclinPB);
        rule26a_1 = min (rule26_1, ErrorWipolPS);

        rule27_1 = min (ErrorAlkohol_1PB, ErrorBayclinPB);
        rule27a_1 = min (rule27_1, ErrorWipolPB);
}
```

Proses *defuzzyfikasi* merupakan tahap akhir dalam pengolahan kontroler *Fuzzy Logic Controler* yang digunakan dalam penelitian kali ini yaitu metode Sugeno. Proses ini menggunakan prinsip *Weight of Average* (WOA) dengan rumus membagi jumlah dari perkalian antara *rule* dengan nilai keluaran dengan jumlah *rule.* Penerapan dalam proses *defuzzyfikasi* bisa dilihat sebagai berikut untuk proses disinfektan :

```
        void Defuzzyfikasi() {
        Rule();
        //OUTPUT Volume Alkohol (dalam satuan %)
        float BKA_1 = 0;
        float BSA_1 = 25;
        float BPA_1 = 50;
        //Output Volume Bayclin (dalam satuan %)
```

```
float BKB = 0;
float BSB = 50;
float BPB = 100;
//OUTPUT Volume Wipol (dalam satuan %)
float BKW = 0;
float BSW = 50;
float BPW = 100;

VolumeAlkohol_1 = (rule1a_1*BKA_1 + rule2a_1*BKA_1 +
rule3a_1*BKA_1 + rule4a_1*BKA_1 + rule5a_1*BKA_1 +
rule6a_1*BKA_1 + rule7a_1*BKA_1 + rule8a_1*BKA_1 +
rule9a_1*BKA_1 + rule10a_1*BSA_1 + rule11a_1*BSA_1 +
rule12a_1*BSA_1 + rule13a_1*BSA_1 + rule14a_1*BSA_1 +
rule15a_1*BSA_1 + rule16a_1*BSA_1 + rule17a_1*BSA_1 +
rule18a_1*BSA_1 + rule19a_1*BPA_1 + rule20a_1*BPA_1 +
rule21a_1*BPA_1 + rule22a_1*BPA_1 + rule23a_1*BPA_1 +
rule24a_1*BPA_1 + rule25a_1*BPA_1 + rule26a_1*BPA_1 +
rule27a_1*BPA_1)/(rule1a_1 + rule2a_1 + rule3a_1 + rule4a_1
+ rule5a_1 + rule6a_1 + rule7a_1 + rule8a_1 + rule9a_1 +
rule10a_1 + rule11a_1 + rule12a_1 + rule13a_1 + rule14a_1 +
rule15a_1 + rule16a_1 + rule17a_1 + rule18a_1 + rule19a_1 +
rule20a_1 + rule21a_1 + rule22a_1 + rule23a_1 + rule24a_1 +
rule25a_1 + rule26a_1 + rule27a_1);

VolumeBayclin = (rule1a_1*BKB + rule2a_1*BKB + rule3a_1*BKB
+ rule4a_1*BSB + rule5a_1*BSB + rule6a_1*BSB + rule7a_1*BPB
+ rule8a_1*BPB + rule9a_1*BPB + rule10a_1*BKB + rule11a_1*BKB
+ rule12a_1*BKB + rule13a_1*BSB + rule14a_1*BSB +
rule15a_1*BSB + rule16a_1*BPB + rule17a_1*BPB + rule18a_1*BPB
+ rule19a_1*BKB + rule20a_1*BKB + rule21a_1*BKB +
rule22a_1*BSB + rule23a_1*BSB + rule24a_1*BSB + rule25a_1*BPB
+ rule26a_1*BPB + rule27a_1*BPB)/(rule1a_1 + rule2a_1 +
rule3a_1 + rule4a_1 + rule5a_1 + rule6a_1 + rule7a_1 +
rule8a_1 + rule9a_1 + rule10a_1 + rule11a_1 + rule12a_1 +
rule13a_1 + rule14a_1 + rule15a_1 + rule16a_1 + rule17a_1 +
rule18a_1 + rule19a_1 + rule20a_1 + rule21a_1 + rule22a_1 +
rule23a_1 + rule24a_1 + rule25a_1 + rule26a_1 + rule27a_1);
```

```
VolumeWipol = (rule1a_1*BKW + rule2a_1*BSW + rule3a_1*BPW +
rule4a_1*BKW + rule5a_1*BSW + rule6a_1*BPW + rule7a_1*BKW +
rule8a_1*BSW + rule9a_1*BPW + rule10a_1*BKW + rule11a_1*BSW
+ rule12a_1*BPW + rule13a_1*BKW + rule14a_1*BSW +
rule15a_1*BPW + rule16a_1*BKW + rule17a_1*BSW + rule18a_1*BPW
+ rule19a_1*BKW + rule20a_1*BSW + rule21a_1*BPW +
rule22a_1*BKW + rule23a_1*BSW + rule24a_1*BPW + rule25a_1*BKW
+ rule26a_1*BSW + rule27a_1*BPW)/(rule1a_1 + rule2a_1 +
rule3a_1 + rule4a_1 + rule5a_1 + rule6a_1 + rule7a_1 +
rule8a_1 + rule9a_1 + rule10a_1 + rule11a_1 + rule12a_1 +
rule13a_1 + rule14a_1 + rule15a_1 + rule16a_1 + rule17a_1 +
rule18a_1 + rule19a_1 + rule20a_1 + rule21a_1 + rule22a_1 +
rule23a_1 + rule24a_1 + rule25a_1 + rule26a_1 + rule27a_1);
}

void setup() {
Serial.begin(57600);
}
void loop() {
  FuzzyfikasiErrorAlkohol_1();
  FuzzyfikasiErrorWipol();
  FuzzyfikasiErrorBayclin();
  Rule();
  Defuzzyfikasi();}
```

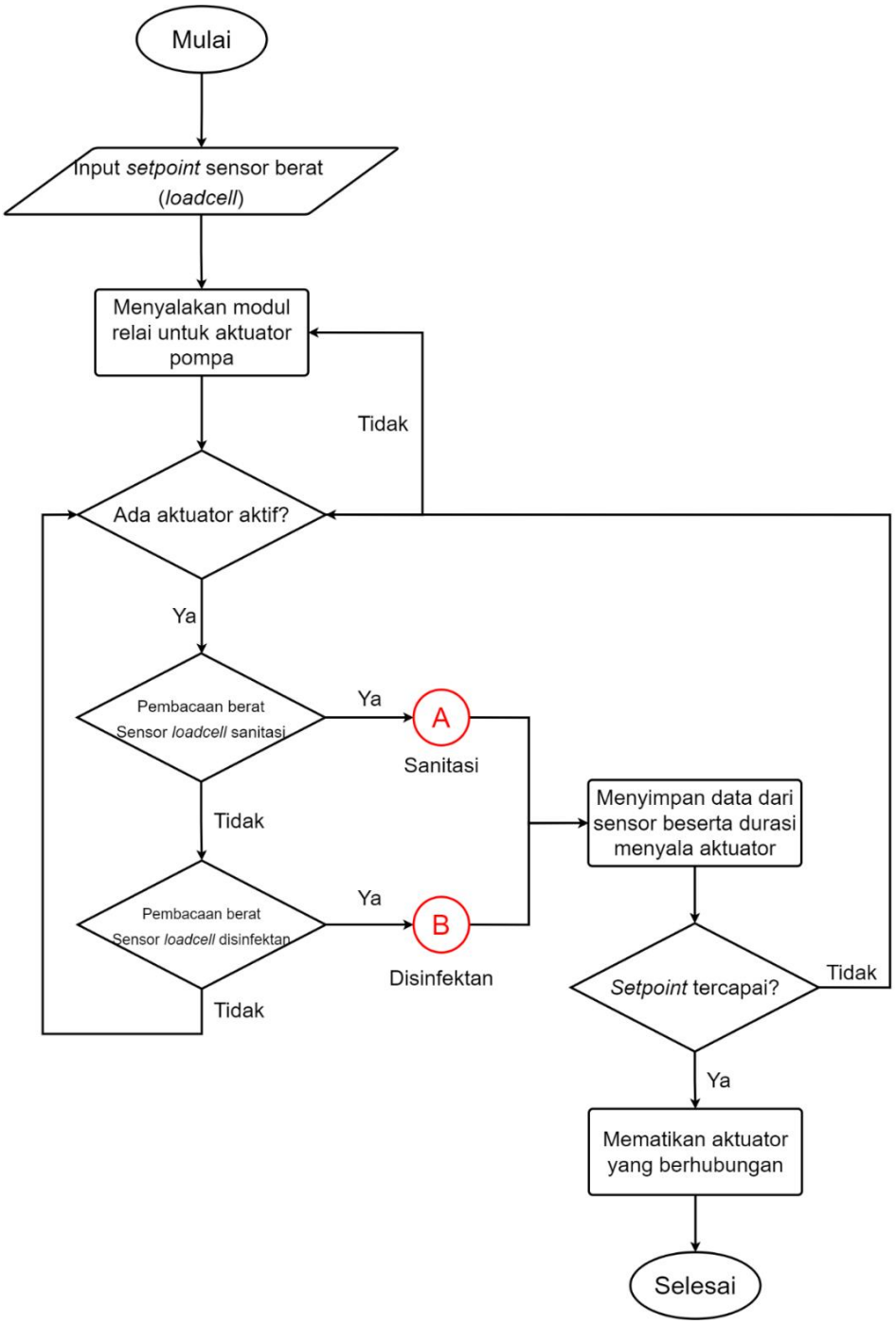
3.5 Sistem Kerja Alat

3.5.1 Sistem Kerja *Prototype* Alat

Berdasarkan Gambar 3.3 dan Gambar 3.4 dapat diketahui sistem kerja *prototype* untuk alat dispenser otomatis yang dikontrol menggunakan modul relai untuk menyalakan aktuator yaitu pompa air DC, berserta sensor *loadcell* yang bekerja sebagai *trigger* batas minimum dan maksimum, dan modul HX711 sebagai penguat pembacaan sensor *loadcell*.

3.5.2 *Flowchart* Sistem Kerja Pembacaan Sensor *Loadcell*

Berikut untuk Gambar 3.21 adalah *flowchart* cara kerja sistem pembacaan sensor *loadcell*.

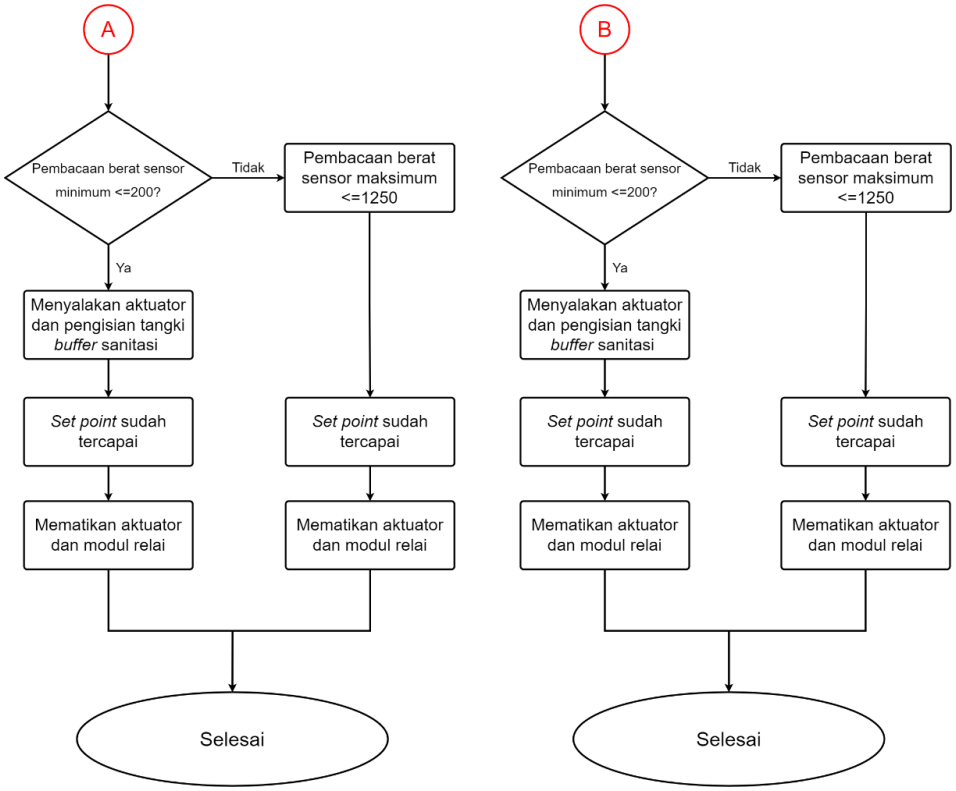


Gambar 3.21. *Flowchart* sistem kerja sensor *loadcell*

Penjelasan *flowchart* sistem kerja pembacaan sensor *loadcell* pada Gambar 3.21 adalah sebagai berikut :

1. Dimulai dengan memasukkan *setpoint* yang sudah diatur ke dalam sensor *loadcell.*
2. Selanjutnya menyalakan modul relai pada aplikasi Arduino untuk mengaktifkan aktuator pompa.
3. Selanjutnya meneruskan sistem apakah ada aktuator yang aktif atau belum, jika belum ada aktuator yang aktif maka akan kembali untuk menyalakan modul relai yang akan mencoba untuk mengaktifkan aktuatornya.
4. Selanjutnya apabila sistem perancangan alat dispenser otomatis ini sudah dapat berjalan, maka dapat dilakukan pembacaan berat pada sensor *load cell*, pada pembacaan ini dibagi menjadi dua bagian untuk pembacaan pada proses sanitasi dan disinfektan, yang akan dijelaskan sesuai dengan Gambar 3.23.
5. Setelah pembacaan sensor *load cell* ini sudah berhasil kemudian data dari sensor berat ini akan tersimpan pada EEPROM yang akan mengatur durasi nyalanya aktuator.
6. Selanjutnya akan masuk ke sistem apakah sudah tercapai sesuai dengan *set point* yang sudah ditentukan oleh peneliti atau belum. Jika belum, maka proses akan kembali menuju opsi apakah aktuator sudah aktif.
7. Selanjutnya setelah sistem sudah mencapai *set point* yang ditentukan, mematikan aktuator yang berhubungan pada sanitasi dan disinfektan.
8. Selesai

Berdasarkan Gambar 3.21 untuk pembacaan berat dari sensor *load cell* ini terbagi menjadi dua bagian, dimana terdapat pembacaan saat di titik minimum atau saat sensor membaca berat tabung *buffer* sebesar $\leq$ 200 gram maka yang terjadi adalah aktuator untuk pengisian tangki *buffer* akan menyala sampai menyentuh titik maksimum pembacaan sensor *load cell* yaitu sebesar $\leq$ 1250 gram, yang dapat dilihat pada Gambar 3.22



Gambar 3.22. *Flowchart* sistem kerja lanjutan

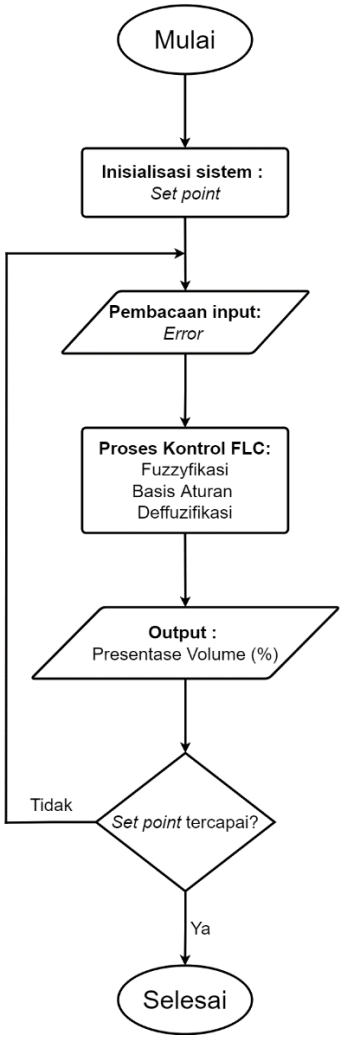
Penjelasan *flowchart* yang berikut ini sesuai dengan Gambar 3.22 dimana pada pembacaan sensor berat atau *load cell* terbagi menjadi dua yaitu proses a dan b, dijelaskan sebagai berikut :

1. Untuk pembacaan sensor berat pada bagian A itu adalah tabung *buffer* sanitasi dibagi menjadi dua opsi ketika pembacaan sensor sudah atau belum mencapai minimum yaitu ≤ 200 gram, sebagai berikut :
   a. Ketika sensor sudah mencapai minimum pembacaan, maka akan dinyalakan aktuator serta dilakukan pengisian tangki *buffer*, lalu saat *set point* sudah tercapai sistem akan mematikan aktuator dan selesai.
   b. Sedangkan ketika pembacaan berat sensor mencapai maksimum yaitu ≤ 1250 gram, aktuator akan mati dan proses selesai.
2. Untuk pembacaan sensor berat pada bagian B itu adalah tabung *buffer* disinfektan yang terbagi menjadi dua pilihan, sebagai berikut :

a. Ketika sensor sudah mencapai minimum pembacaan, maka akan dinyalakan aktuator serta dilakukan pengisian tangki *buffer*, lalu saat *set point* sudah tercapai sistem akan mematikan aktuator dan selesai.
b. Sedangkan ketika pembacaan berat sensor mencapai maksimum yaitu ≤ 1250 gram, aktuator akan mati dan proses selesai.

3.5.3 *Flowchart* Sistem Kerja *Fuzzy Logic Control*

Berikut untuk Gambar 3.23 adalah *flowchart* kerja *Fuzzy Logic Control* pada sistem dispenser otomatis.



Gambar 3.23. *Flowchart Fuzzy Logic Control*

Penjelasan *flowchart* Gambar 3.23 sistem *Fuzzy Logic* sebagai berikut :

1. Dimulai dengan menginisialisasi sistem berupa *set point*
2. Kemudian proses selanjutnya adalah pembacaan *input* berupa *error* pada tiap tabung bahan sanitasi maupun disinfektan.
3. Selanjutnya *input* yang telah terbaca akan diseleksi berdasarkan proses kontrol FLC, yang dimulai dengan pemrosesan *Fuzzyfication* terlebih dahulu yang diseleksi berdasarkan himpunan, kemudian ada basis aturan FLC, dan *defuzzyfication.*
4. Lalu *output* dari sistem FLC ini yaitu berupa presentase volume (%).
5. Setelahnya apakah *set point* sudah tercapai atau belum. Ketika belum sesuai dengan *set point* yang dituju maka proses akan dikembalikan pada proses antara inisialisasi sistem dan pembacaan *input*, sedangkan jika sudah tercapai maka sistem langsung selesai.

BAB IV
PENGUJIAN DAN ANALISIS

Pada bab ini membahas mengenai pengujian dan analisis sistem perancangan MCU pada pengaturan volume cairan menggunakan *Fuzzy Logic* metode Sugeno dari pengujian yang meliputi pengukuran dan pembacaan sensor *load cell*, dan pengujian pada pengambilan keputusan *error* untuk tiap bahan dan volume sebagai *output*.

4.1 Pengujian Pada Pengukuran Pembacaan Sensor *Load Cell*

Pengujian untuk pengukuran sensor berat ini dilakukan dengan adanya batasan minimum dan batasan maksimum untuk mengaktifkan aktuator bekerja aktif menggunakan relai. Pengujian ini dilakukan setiap 2 menit sekali sampai pada menit ke 10 yang terbaca pada sensor dan terdapat sebuah *error* yang berbeda setiap menitnya dibandingkan dengan tangki *buffer* yang menggunakan gelar takar seberat 1 liter. *Error* dapat dilihat seperti pada tabel 4.1.

Tabel 4.1. Pengukuran Berat Batas Minimum Menggunakan Sensor *Loadcell*

No.	Pembacaan Sensor Batas Minimum (gram)		Gelas Ukur (gram)		Error (%)		Pembacaan saat menit ke -
	Tangki Sanitasi	Tangki Disinfektan	Tangki Sanitasi	Tangki Disinfektan	Tangki Sanitasi	Tangki Disinfektan	
1.	464,30	374,43	450	370	14,30	4,43	Menit ke 2
2.	464,12	374,30	450	370	14,12	4,30	Menit ke 4
3.	463,92	374,24	450	370	13,92	4,24%	Menit ke 6
4.	463,65	373,81	450	370	13,65	3,81	Menit ke 8
5.	463,49	374,32	450	370	13,49	4,32	Menit ke 10

Pada Tabel 4.1 dilakukan pengujian batas minimum pembacaan sensor *loadcell* selama 10 menit dan diamati ketika 2 menit sekali untuk menentukan adanya *error* pada pengujian. Bisa dilihat pada tabel diatas terdapat *error* yang cukup besar untuk sanitasi sebesar 14,30% saat pembacaan menit pertama, sedangkan pada pembacaan menit pertama *error* tangki disinfektan kurang dari 5% yaitu 4,43%. Begitupula untuk menit seterusnya selama 2 menit sekali, pada pembacaan *error* tangki sanitasi terbaca 14,12%, 13,92%, 13,65%, dan 13,49%. Sedangkan untuk setiap 2 menit berikutnya *error* tangki disinfektan terbaca 4,30%, 4,24%, 3,82%, dan 4,32%.

Tabel 4.2. Pengukuran Berat Batas Maksimum Menggunakan Sensor *Loadcell*

No.	Pembacaan Sensor Batas Maksimum (gram)		Gelas Ukur (gram)		Error		Pembacaan saat menit ke -
	Tangki Sanitasi	Tangki Disinfektan	Tangki Sanitasi	Tangki Disinfektan	Tangki Sanitasi	Tangki Disinfektan	
1.	941,68	961,48	930	950	11,68	11,48	Menit ke 2
2.	941,61	961,50	930	950	11,61	11,50	Menit ke 4
3.	941,54	961,68	930	950	11,54	11,68	Menit ke 6
4.	941,44	961,32	930	950	11,44	11,32	Menit ke 8
5.	941,30	961,64	930	950	11,30	11,64	Menit ke 10

Pada Tabel 4.2 dilakukan pengujian batas maksimum pembacaan sensor *loadcell* selama 10 menit dan diamati ketika 2 menit sekali untuk menentukan adanya *error* pada pengujian. Bisa dilihat pada tabel diatas terdapat *error* yang cukup besar untuk sanitasi sebesar 11,68% saat pembacaan menit pertama, sedangkan pada pembacaan menit pertama *error* tangki disinfektan memiliki pembacaan yang tidak terlalu signifikan jauh yaitu 11,48%. Begitupula untuk menit seterusnya selama 2 menit sekali, pada pembacaan *error* tangki sanitasi terbaca 11,61%, 11,54%, 11,44%, dan 11,30%. Sedangkan untuk setiap 2 menit berikutnya *error* tangki disinfektan terbaca 11,50%, 11,68%, 11,32%, dan 11,64%.

4.2 Pengujian *Fuzzy Logic Control* Pada *Volume* Cairan

Logika *Fuzzy Logic* digunakan untuk mengendalikan volume berdasarkan komposisi masing – masing tabung bahan yaitu *alkohol sanitasi, glycerin, H_2O_2, alkohol disinfektan, wipol, dan bayclin.* Variabel yang digunakan untuk logika *Fuzzy* adalah nilai *error* setiap tabung bahan yang didapat dari selisih antara nilai *set point* yang telah ditetapkan oleh peneliti. Dari variabel tersebut akan digunakan untuk menentukan besar persentase volume pada hasil *output*. Pada pengujian logika *Fuzzy Logic* pada Tugas Akhir ini menggunakan metode Sugeno. Pengujian ini dilalukan dengan cara menghitung besar volume yang didapatkan oleh perhitungan logika *Fuzzy Logic* secara manual dengan hasil nilai nilai volume sesuai program. Perhitungan ini digunakan untuk membandingkan hasil volume dalam perhitungan dan volume yang didapatkan oleh sistem.

Tabel 4.3. Pengujian *error* sanitasi

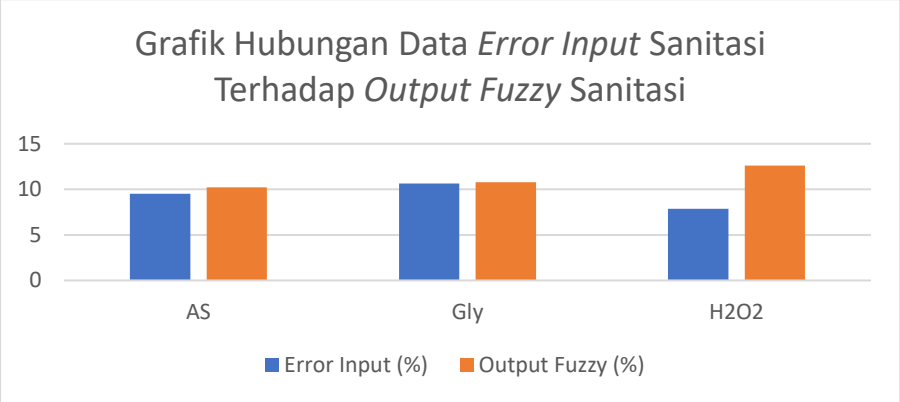
No	Bahan	*Set point* (%)	*Error* (%)
1	Alkohol Sanitasi	15	9,50
2	Glycerin	30	10,65
3	H_2O_2	46	7,88

Berdasarkan data pada Tabel 4.3 dapat diketahui nilai *set point* yang digunakan diambil secara acak. Namun, masih berdasarkan *rule Fuzzy* yang sudah ditentukan oleh peneliti. Nilai *error* didapatkan dari hasil pengujian (tidak terdapat *feedback* karena *open loop*). Pada nilai *error* ini adalah nilai yang absolut, jadi nilai *error* tersebut digunakan sebagai masukan *Fuzzy* untuk menghasilkan volume yang tepat dalam proses pemcampuran sanitasi. Dari tabel diatas didapatkan nilai rata – rata *error* untuk AS (Alkohol Sanitasi) sebanyak 17,50%, *error* Gly (Glycerin) sebanyak 22,65%, dan *error* H_2O_2 sebanyak 24,61%. Data *error* yang besar ini disebabkan karena standar pencampuran yang digunakan oleh peneliti berbeda dengan standar pencampuran sesungguhnya. Hasil pembacaan kontroler *Fuzzy Logic Control* metode Sugeno dapat dilihat pada Tabel 4.4.

Tabel 4.4. Hasil pengujian kontroler sanitasi

No	Bahan	*Error input* (%)	*Output Fuzzy* (%)	Volume (ml)
1	AS	9,50	10,20	12,80
2	Gly	10,65	10,78	15,23
3	H_2O_2	7,88	12,61	12,59

Berdasarkan data pada tabel 4.4 dapat diketahui rata – rata nilai keluaran *Fuzzy* untuk Alkohol Sanitasi (AS) sebesar 10,20%, Glycerin (Gly) sebesar 10,78%, dan H_2O_2 sebesar 12,61%. Sedangkan untuk rata- rata volume Alkohol Sanitasi (AS) sebesar 12,80 ml, Glycerin (Gly) sebesar 15,23 ml, dan H_2O_2 sebesar 12,59 ml.



Gambar 4.1. Grafik hubungan data *error input* sanitasi terhadap *output Fuzzy* sanitasi

Berdasarkan pada Gambar 4.1 dapat diketahui bahwa semakin besar *error input* maka presentase volume yang dihasilkan akan semakin besar pula. Maka dari hal tersebut dapat disimpulkan bahwa semakin besar nilai *output Fuzzy* maka volume yang dihasilkan akan semakin besar, dan semakin kecil nilai *output Fuzzy* maka volume yang dihasilkan juga semakin kecil.

Tabel 4.5. Pengujian error disinfektan

No	Bahan	*Set point* (%)	*Error* (%)
1	Alkohol Disinfektan	20	10,50
2	Bayclin	50	14,23
3	Wipol	35	9,76

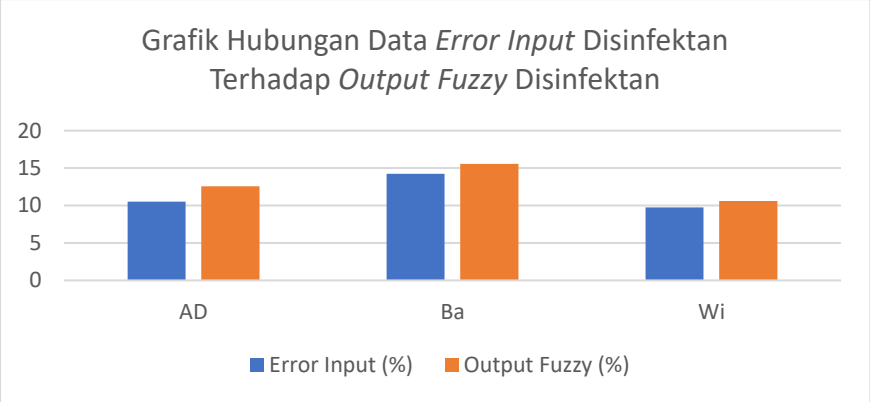
Berdasarkan data pada Tabel 4.5 dapat diketahui nilai *set point* yang digunakan diambil secara acak. Namun, masih berdasarkan *rule Fuzzy* yang sudah ditentukan oleh peneliti. Nilai *error* didapatkan dari hasil pengujian (tidak terdapat *feedback* karena sistem *Fuzzy open loop*). Pada nilai *error* ini adalah nilai yang absolut, jadi nilai *error* tersebut digunakan sebagai masukan *Fuzzy* untuk menghasilkan volume yang tepat dalam proses pemcampuran sanitasi. Dari tabel diatas didapatkan nilai rata – rata *error* untuk AD (Alkohol Disinfektan) sebanyak 10,50%, *error* Ba (Bayclin) sebanyak 14,23%, dan *error* Wi (Wipol) sebanyak 9,76%. Data *error* yang besar ini disebabkan karena standar pencampuran yang digunakan oleh peneliti berbeda dengan standar pencampuran sesungguhnya. Hasil pembacaan kontroler *Fuzzy Logic Control* metode Sugeno dapat dilihat pada Tabel 4.6.

Tabel 4.6. Hasil pengujian kontroler disinfektan

No	Bahan	*Error input* (%)	*Output Fuzzy* (%)	Volume (ml)
1	AD	10,50	12,56	40,76
2	Ba	14,23	15,58	53,35
3	Wi	9,76	10,61	37,89

Berdasarkan data pada tabel 4.2 dapat diketahui rata – rata nilai keluaran *Fuzzy* untuk Alkohol Sanitasi (AS) sebesar 12,56%, Glycerin (Gly) sebesar 15,58%, dan H_2O_2 sebesar 10,61%. Sedangkan untuk rata- rata volume Alkohol

Sanitasi (AS) sebesar 40,76 ml, Glycerin (Gly) sebesar 53,35 ml, dan H_2O_2 sebesar 37,89 ml.

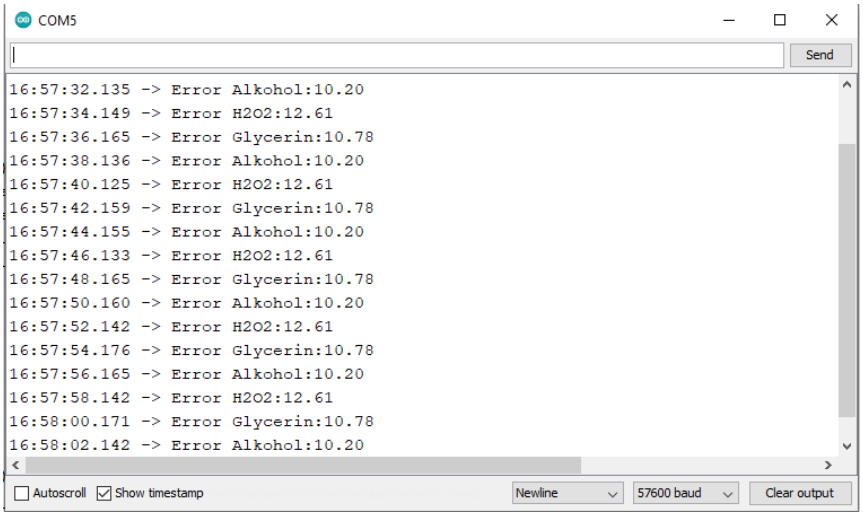


Gambar 4.2. Grafik hubungan data *error input* disinfektan terhadap *output Fuzzy* disinfektan

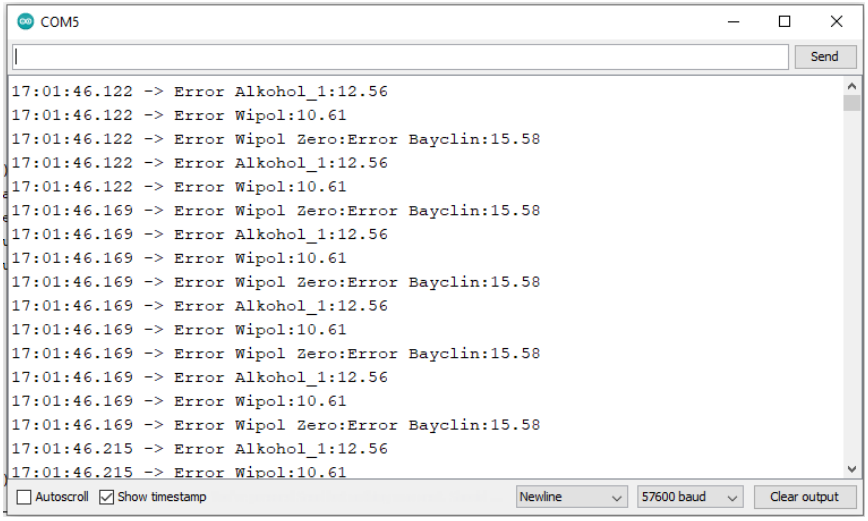
Berdasarkan pada Gambar 4.2 dapat diketahui bahwa semakin besar *error input* maka presentase volume yang dihasilkan akan semakin besar pula. Maka dari hal tersebut dapat disimpulkan bahwa semakin besar nilai *output Fuzzy* maka volume yang dihasilkan akan semakin besar, dan semakin kecil nilai *output Fuzzy* maka volume yang dihasilkan juga semakin kecil.

4.2.1 Perbandingan Perhitungan *Fuzzy* Dengan Hasil Pengujian Pada Alat Dispenser Otomatis

Dilakukan perhitungan *Fuzzy* secara manual untuk membandingkan hasil perhitungan *Fuzzy* dengan hasil pengujian pada alat dispenser otomatis. *Range set point* pada Tugas Akhir ini yaitu sebesar 0 hingga 100%. Perhitungan *Fuzzy* menggunakan nilai yang absolut.



Gambar 4.3. Hasil pembacaan *error* cairan sanitasi



Gambar 4.4. Hasil pembacaan *error* cairan desinfektan

Berikut ini perhitungan menggunakan 6 sampel bahan dengan menggunakan logika *Fuzzy*.

1. Perhitungan Pada Alkohol Sanitasi

   Berdasarkan konsep logika *Fuzzy*, terlebih dahulu dihitung nilai *error* dengan perhitungan yang sudah disesuaikan oleh peneliti melalui simulink dan Arduino IDE. Untuk nilai *error input* didapatkan sebesar 9,50 (*Zero*), sedangkan untuk *error Fuzzy* nya terbaca 10,20%.

2. Perhitungan Pada Glycerin

Berdasarkan konsep logika *Fuzzy*, terlebih dahulu dihitung nilai *error* dengan perhitungan yang sudah disesuaikan oleh peneliti melalui simulink dan Arduino IDE. Untuk nilai *error input* didapatkan sebesar 10,65 (PS), sedangkan untuk *error Fuzzy* nya terbaca 10,78%.

3. Perhitungan Pada H_2O_2

   Berdasarkan konsep logika *Fuzzy*, terlebih dahulu dihitung nilai *error* dengan perhitungan yang sudah disesuaikan oleh peneliti melalui simulink dan Arduino IDE. Untuk nilai *error input* didapatkan sebesar 7,88 (PB), sedangkan untuk *error Fuzzy* nya terbaca 12,61%.

4. Perhitungan Pada Alkohol Disinfektan

   Berdasarkan konsep logika *Fuzzy*, terlebih dahulu dihitung nilai *error* dengan perhitungan yang sudah disesuaikan oleh peneliti melalui simulink dan Arduino IDE. Untuk nilai *error input* didapatkan sebesar 10,50 (*Zero*), sedangkan untuk *error Fuzzy* nya terbaca 12,56%.

5. Perhitungan Pada Bayclin

   Berdasarkan konsep logika *Fuzzy*, terlebih dahulu dihitung nilai *error* dengan perhitungan yang sudah disesuaikan oleh peneliti melalui simulink dan Arduino IDE. Untuk nilai *error input* didapatkan sebesar 14,24 (PS), sedangkan untuk *error Fuzzy* nya terbaca 15,58%.

6. Perhitungan Pada Wipol

   Berdasarkan konsep logika *Fuzzy*, terlebih dahulu dihitung nilai *error* dengan perhitungan yang sudah disesuaikan oleh peneliti melalui simulink dan Arduino IDE. Untuk nilai *error input* didapatkan sebesar 9,76 (PB), sedangkan untuk *error Fuzzy* nya terbaca 10,61%.

4.3 Pengujian Alat Dispenser Otomatis Secara Keseluruhan

Pada pengujian alat keseluruhan perancangan dispenser otomatis akan diujikan terhadap hasil akhir dari keluaran sistem. Terdapat masing – masing 3 keluaran untuk cairan dalam besaran mililiter, pengujian ini akan berfokus pada pengujian proses pengisian sanitasi maupun desinfektan terhadap *user* atau

pengguna dimana akan menerima hasil yang sesuai dengan aplikasi sebanyak 100ml, 150ml, dan 200ml.

A. Pengujian saat 100ml pada kedua cairan.

Pada pengujian kali ini, telah dilakukan secara keseluruhan untuk memproses kedua cairan yaitu sanitasi maupun desinfektan sebanyak 100ml dalam sekali percobaan. Gambar 4.3 dan Gambar 4.4 merupakan bukti bahwa pengujian untuk 100ml cairan sanitasi maupun desinfektan telah berhasil.

Gambar 4.5. Pengujian 100ml cairan sanitasi

Pada gambar diatas merupakan pengujian cairan sanitasi saat keluaran dalam sistem sebanyak 100ml pengujian kedua pada Tabel 4.7.

Tabel 4.7. Pengujian 100ml cairan sanitasi

Pengujian ke -	Pengujian yang terukur menggunakan gelas ukur (ml)	*Set point* (%)	*Error* (%)
1	90	100	10
2	100	100	0
3	120	100	20
4	100	100	0

Pada Tabel 4.7 menunjukkan bahwa penguji melakukan pengujian keseluruhan sebanyak empat kali, dan pada pengujian ke – 2 terlihat bahwa

tidak terdapatnya *error* dikarenakan keluaran aliran cairan yang tidak selalu sama

Gambar 4.6. Pengujian 100ml cairan desinfektan

Pada gambar diatas merupakan pengujian cairan desinfektan saat keluaran dalam sistem sebanyak 100ml untuk pengujian ketiga dan keempat pada Tabel 4.8.

Tabel 4.8. Pengujian 100ml cairan desinfektan

Pengujian ke -	Pengujian yang terukur menggunakan gelas ukur (ml)	*Set point* (%)	*Error* (%)
1	90	100	10
2	110	100	10
3	100	100	0
4	100	100	0

Pada Tabel 4.8 menunjukkan bahwa penguji melakukan pengujian keseluruhan sebanyak empat kali, dan pada pengujian ketiga dan keempat terlihat bahwa tidak terdapatnya *error* dikarenakan keluaran aliran cairan yang tidak selalu sama.

B. Pengujian saat 150ml pada kedua cairan.

Pada pengujian kali ini, telah dilakukan secara keseluruhan untuk memproses kedua cairan yaitu sanitasi maupun desinfektan sebanyak 150ml dalam sekali percobaan. Gambar 4.5 dan Gambar 4.6 merupakan bukti bahwa pengujian untuk 150ml cairan sanitasi maupun desinfektan telah berhasil.

Gambar 4.7. Pengujian 150ml cairan sanitasi

Pada gambar diatas merupakan pengujian cairan sanitasi saat keluaran dalam sistem sebanyak 170ml pada pengujian kedua dalam Tabel 4.9.

Tabel 4.9. Pengujian 150ml cairan sanitasi

Pengujian ke -	Pengujian yang terukur menggunakan gelas ukur (ml)	*Set point* (%)	*Error* (%)
1	160	150	10
2	170	150	20
3	180	150	30
4	150	150	0

Pada Tabel 4.9 menunjukkan bahwa penguji melakukan pengujian keseluruhan sebanyak empat kali, dan pada pengujian ke – 4 terlihat bahwa

tidak terdapatnya *error* dikarenakan keluaran aliran cairan yang tidak selalu sama.

Gambar 4.8. Pengujian 150ml cairan desinfektan

Pada gambar diatas merupakan pengujian cairan desinfektan saat keluaran dalam sistem sebanyak 150ml pada pengujian ketiga dalam Tabel 4.10.

Tabel 4.10. Pengujian 150ml cairan desinfektan

Pengujian ke -	Pengujian yang terukur menggunakan gelas ukur (ml)	*Set point* (%)	*Error* (%)
1	140	150	10
2	160	150	10
3	150	150	0
4	190	150	40

Pada Tabel 4.10 menunjukkan bahwa penguji melakukan pengujian keseluruhan sebanyak empat kali, dan pada pengujian ke – 3 terlihat bahwa tidak terdapatnya *error* dikarenakan aliran cairan yang tidak selalu sama untuk keluarannya.

C. Pengujian saat 200ml pada kedua cairan.

Pada pengujian kali ini, telah dilakukan secara keseluruhan untuk memproses kedua cairan yaitu sanitasi maupun desinfektan sebanyak 200ml

dalam sekali percobaan. Gambar 4.7 dan Gambar 4.8 merupakan bukti bahwa pengujian untuk 200ml cairan sanitasi maupun desinfektan telah berhasil.

Gambar 4.9. Pengujian 200ml cairan sanitasi

Pada gambar diatas merupakan pengujian cairan sanitasi saat keluaran dalam sistem sebanyak 220ml.

Tabel 4.11. Pengujian 200ml cairan sanitasi

Pengujian ke -	Pengujian yang terukur menggunakan gelas ukur (ml)	*Set point* (%)	*Error* (%)
1	220	200	20
2	230	200	30
3	200	200	0
4	210	200	10

Pada Tabel 4.11 menunjukkan bahwa penguji melakukan pengujian keseluruhan sebanyak empat kali, dan pada pengujian ke – 3 terlihat bahwa tidak terdapatnya *error* dikarenakan aliran cairan yang tidak selalu sama untuk keluarannya.

Gambar 4.10. Pengujian 200ml cairan desinfektan

Pada gambar diatas merupakan pengujian cairan desinfektan saat keluaran dalam sistem sebanyak 200ml.

Tabel 4.12. Pengujian 200ml cairan desinfektan

Pengujian ke -	Pengujian yang terukur menggunakan gelas ukur (ml)	*Set point* (%)	*Error* (%)
1	200	200	0
2	210	200	10
3	220	200	20
4	200	200	0

Pada Tabel 4.12 menunjukkan bahwa penguji melakukan pengujian keseluruhan sebanyak empat kali, dan pada pengujian pertama dan keempat terlihat bahwa tidak terdapatnya *error* dikarenakan aliran cairan yang tidak selalu sama untuk keluarannya.

BAB V
PENUTUP

5.1 Kesimpulan

Berdasarkan hasil pengujian dan analisis yang telah dilakukan, maka dapat diambil kesimpulan sebagai berikut :

1. Telah berhasilnya dirancang alat pengisian dispenser otomatis dengan terdapatnya *error* 10% dan 20% pada saat proses pengujian cairan sanitasi untuk 100ml dalam tabung ukur. Terdapat *error* yang paling tinggi sebesar 40% pada saat proses pengujian cairan desinfektan 150ml pada pengujian keempat.
2. Diperoleh pada saat pengujian dan pengukuran sensor berat menggunakan sensor *load cell* untuk batas maksimum tabung sanitasi terdapat *error* sebesar 11,68% pada pembacaan awal yaitu menit ke – 2, kemudian saat dilakukan pembacaan menit ke – 4 terdapat *error* sebesar 11,61%, selanjutnya pada pembacaan menit ke – 6 terdapat *error* sebesar 11,54%, selanjutnya pada pembacaan menit ke – 8 terdapat *error* yang semakin menurun yaitu sebesar 11,44%, dan terakhir pada pembacaan menit ke – 10 terdapat *error* sebesar 11,30%.
3. Diperoleh pada saat pengujian dan pengukuran sensor berat menggunakan sensor *load cell* untuk batas maksimum tabung disinfektan terdapat *error* sebesar 11,48% pada pembacaan awal yaitu menit ke – 2, kemudian saat dilakukan pembacaan menit ke – 4 terdapat *error* sebesar 11,50%, selanjutnya pada pembacaan menit ke – 6 terdapat *error* sebesar 11,68%, selanjutnya pada pembacaan menit ke – 8 terdapat *error* yang semakin menurun yaitu sebesar 11,32%, dan terakhir pada pembacaan menit ke – 10 terdapat *error* sebesar 11,64%.

4. Untuk hasil pembacaan *error Fuzzy Logic* bahan sanitasi sebesar 10,20% untuk alkohol, 10,78% untuk glycerin, dan H_2O_2 sebesar 12,61% dan didapatkan juga volume sebanyak 12,80 ml untuk alkohol sanitasi, 15,23 ml untuk glycerin, dan 12,59 ml untuk H_2O_2.
5. Untuk hasil pembacaan *error Fuzzy Logic* bahan sanitasi sebesar 12,56% untuk alkohol, 15,58% untuk bayclin, dan wipol sebesar 10,61% dan didapatkan juga volume sebanyak 40,76 ml untuk alkohol disinfektan, 53,35 ml untuk bayclin, dan 37,89 ml untuk wipol.
6. Telah berhasil dirancang protoripe alat dispenser otomatis dengan sistem *Fuzzy Logic Control* yang menghasilkan cairan sanitasi dan disinfektan sebanyak 100ml, 150ml, dan 200ml.

5.2 Saran

Saran untuk pengembangan pada penelitian selanjutnya, yaitu :

1. Lebih dipertimbangkan lagi untuk keakuratan keluaran pada setiap cairan dimulai dari tabung bahan, tabung *buffer*, sampai dengan ke pengguna / *user*.
2. Mengembangkan metode *Fuzzy Logic Control* dengan metode yang sama atau berbeda yang dimana dapat menambahkan jumlah masukan, sehingga diharapkan sistem dapat bekerja dengan lebih akurat atau presisi.
3. Dapat diteliti lebih lanjut mengenai bahan – bahan olahan dasar untuk sanitasi dan disinfektan.
4. Menggunakan batas maksimum atau minimum yang lebih besar maupun kecil, dan menggunakan sensor berat yang sama atau berbeda.

DAFTAR PUSTAKA

[1] A. G. A. N. İ, Ö. F. K. İ. O. Ğ. Lu, H. Açikgöz, and Ş. İ. Ekkel, "Fuzzy Logic Controller Design Based on Sugeno Inference Method for Nonlinear Inverted Pendulum Dynamical System," vol. 8, no. 1, pp. 19–30, 2017.

[2] Y. Zamrodah, "済無No Title No Title No Title," vol. 15, no. 2, pp. 1–23, 2016.

[3] B. Firman, "Implementasi Sensor IMU MPU6050 Berbasis Serial I2C Pada Self-Balancing Robot Vol . 9 No . 1 Agustus 2016 ISSN : 1979-8415," *Juenal Teknol. Technoscientia*, vol. 9, no. 1, pp. 18–24, 2016.

[4] M. Maa, "Balancing Robot Beroda Dua Menggunakan Metoda Kontrol Proporsional, Integral dan Derivatif," *J. Elektro dan Mesin Terap.*, vol. 2, no. 1, pp. 34–42, 2016, doi: 10.35143/elementer.v2i1.48.

[5] X. Liu, H. Fang, and Z. Chen, "A novel cost function based on decomposing least-square support vector machine for Takagi-Sugeno fuzzy system identification," *IET Control Theory Appl.*, vol. 8, no. 5, pp. 338–347, 2014, doi: 10.1049/iet-cta.2013.0707.

[6] R. Z. Homod, K. S. M. Sahari, H. A. F. Almurib, and F. H. Nagi, "Gradient auto-tuned Takagi-Sugeno Fuzzy Forward control of a HVAC system using predicted mean vote index," *Energy Build.*, vol. 49, pp. 254–267, 2012, doi: 10.1016/j.enbuild.2012.02.013.

[7] F. G. Becker *et al.*, "No 主観的健康感を中心とした在宅高齢者における 健康関連指標に関する共分散構造分析Title," *Syria Stud.*, vol. 7, no. 1, pp. 37–72, 2015, [Online]. Available: https://www.researchgate.net/publication/269107473_What_is_governance/link/548173090cf22525dcb61443/download%0Ahttp://www.econ.upf.edu/~reynal/Civil wars_12December2010.pdf%0Ahttps://think-asia.org/handle/11540/8282%0Ahttps://www.jstor.org/stable/41857625.

[8] Robotshop, "Datasheet 3133-Micro Load Cell (0-5kg)-CZL635,"

Robotshop, p. 4, 2011, [Online]. Available: https://www.robotshop.com/media/files/pdf/datasheet-3133.pdf.

[9] Avia, "Data Sheet - HX-711," *Avia Semicond.*, vol. 1, no. 1, pp. 1–9, 2017, [Online]. Available: https://cdn.sparkfun.com/datasheets/Sensors/ForceFlex/hx711_english.pdf.

[10] Y. Makasudede, "Bab 2 tinjauan pustaka," pp. 8–45, 1953.

[11] L. Belakang, "Bab I ْ ي ُ خ ِ حض يا," *Galang Tanjung*, no. 2504, pp. 1–9, 2015.

[12] Atmel, "Inspiring Smart and Secure Connected Designs," vol. 11, p. 28, 2015.

[13] Y. Dote, "Introduction to fuzzy logic," *IECON Proc. (Industrial Electron. Conf.*, vol. 1, no. September 2005, pp. 50–56, 1995, doi: 10.2298/fuee0502319m.

[14] H. Otmane, M. Youssef, and B. Mokhtar, "Comparative analysis of cascaded fuzzy-PI controllers based-mppt and perturb and observe mppt in a grid-connected PV system operating under different weather and loading conditions," *Int. J. Power Electron. Drive Syst.*, vol. 10, no. 4, pp. 1986–1994, 2019, doi: 10.11591/ijpeds.v10.i4.1986-1994.

[15] T. J.ROSS, *Fuzzy Logic With Engineering Application*. 2010.

[16] D. N. Utama, "Logika Fuzzy Untuk Model Penunjang Keputusan," no. July, p. 155, 2021.

[17] B. Wismarizqa, D. Ana, and R. Wati, "DESAIN SISTEM KENDALI FEEDBACK PLUS FEEDFORWARD PADA STIRRED-TANK HEATING PROCESS BERBASIS LABVIEW," pp. 26–32, 2016.

[18] D. Wang and Y. Bai, "Implementation of fuzzy logic control systems," *Adv. Ind. Control*, no. 9781846284687, pp. 37–52, 2006, doi: 10.1007/978-1-84628-469-4_3.

[19] A. P. Jacquin and A. Y. Shamseldin, "Review of the application of fuzzy inference systems in river flow forecasting," *J. Hydroinformatics*, vol. 11, no. 3–4, pp. 202–210, 2009, doi: 10.2166/hydro.2009.038.

[20] D. L. Rahakbauw, "Penerapan Logika Fuzzy Metode Sugeno Untuk

Menentukan Jumlah Produksi Roti Berdasarkan Data Persediaan Dan Jumlah Permintaan," *BAREKENG J. Ilmu Mat. dan Terap.*, vol. 9, no. 2, pp. 121–134, 2015, doi: 10.30598/barekengvol9iss2pp121-134.

[21] G. T. Tulak, S. Ramadhan, and A. Musrifah, "Edukasi Perilaku Cuci Tangan Pakai Sabun Pada Siswa Untuk Pencegahan Transmisi Penyakit," *JMM (Jurnal Masy. Mandiri)*, vol. 4, no. 1, p. 37, 2020, doi: 10.31764/jmm.v4i1.1702.

BIODATA

Nama Mahasiswa	:	Yohanes Juan Kurniadi
NIM	:	21060118140124
Konsentrasi	:	Teknik Kontrol Optimasi
Tempat, Tgl Lahir	:	Tangerang, 14 Juni 1999
Alamat Sekarang	:	Jl. Ngesrep Timur Dalam III No. 8, Banyumanik, Kota Semarang
No. Telp. HP	:	081222743666
E-mail	:	yohanesjuan29@gmail.com
Nama Orang Tua	:	Billy Zukyawan
Alamat Orang Tua	:	Cluster Costarica Blok CK7 No.12, Modernland, Kota Tangerang.
IPK Kumulatif	:	3,50

Pengalaman dan prestasi yang pernah diraih :

1. EDMAT-41 (*Engineering Development, Motivation and Awareness Training*) of University of Malaya.
2. Member of *Electrical Workshop* Universitas Diponegoro.
3. Himpunan Mahasiswa Elektro Undip bidang Pendidikan dan Penalaran.

Semarang, 9 September 2022

Yohanes Juan Kurniadi

21060118140124

LAMPIRAN A
MAKALAH TUGAS AKHIR

MCU Design On Liquid Volume Control Using Fuzzy Logic Controller Sugeno Method

Yohanes Juan Kurniadi[1*)], Ajub Ajulian Zahra, S.T., M.T.[2], Sumardi, S.T., M.T.[2]

[1]Mahasiswa dan [2]Dosen Pembimbing
Program Studi Sarjana Departemen Teknik Elektro, Universitas Diponegoro
Jl. Prof. Soedharto, S.H., Kampus UNDIP Tembalang, Semarang, Jawa Tengah, 50275, Indonesia
[*)] *E-mail* : yohanesjuan29@gmail.com

Abstrak

Dalam upaya untuk membuat pembangunan berkelanjutan di tengah pandemi COVID-19 dibuatlah sebuah Alat & Dispenser Desinfektan dan *Hand Sanitizer* otomatis dengan menanamkan teknologi *touchless* menggunakan *Fuzzy Logic Controller* metode Sugeno sebagai pengampil keputusan. Pada penilitian ini bertujuan untuk mendukung kebiasaan tanpa setuhan. Alat Tugas Akhir ini menggunakan pompa air DC sebagai aktuator yang mengalirkan cairan – cairan dari titik satu ke titik lainnya yang sudah disesuaikan oleh peneliti. Sensor yang digunakan untuk menimbang tangki penampung adalah sensor berat (*loadcell*). Parameter yang akan diambil adalah parameter batas minimum dan maksimum untuk sensor *loadcell*, kemudian hasil *output* dari kontroler *Fuzzy Logic*. Diperoleh hasil penelitian perhitungan batas minimum untuk *loadcell* pada menit ke – 2 diperolah *error* sebesar 14,30% untuk sanitasi dan menit ke – 2 diperoleh *error* sebesar 4,43% untuk batas disinfektan. Sedangkan batas maksimum, pada menit ke – 2 diperoleh *error* sebesar 11,68% untuk sanitasi dan menit ke – 10 diperoleh *error* sebesar 11,64% untuk disinfektan. Begitupula untuk hasil perhitungan *Fuzzy Logic* yang sudah didapat untuk sanitasi nilai *output* sebesar 10,20% untuk Alkohol Sanitasi, 10,78% untuk Glycerin, dan 12,62% untuk H_2O_2. Begitupula untuk hasil nilai *output* disinfektan sebesar 12,56% untuk Alkohol Disinfektan, 15,58% untuk Bayclin, dan 10,61% untuk Wipol.

***Kata Kunci**: Fuzzy Logic Controller, loadcell, dan Pompa air DC*

Abstract

To create sustainable development during the COVID-19 pandemic, an automatic Disinfectant and Hand Sanitizer Tool & Dispenser was created by embedding touchless technology using the Sugeno Fuzzy Logic Controller method as a decision maker. This research aims to support the habit of no god. This final project tool uses a DC water pump as an actuator that flows liquids from one point to another that has been adjusted by the researcher. The sensor used to weigh the holding tank is a weight sensor (loadcell). The parameters to be taken are the minimum and maximum limit parameters for the loadcell sensor, then the output results from the Fuzzy Logic controller. The results of the research on the calculation of the minimum limit for loadcell in the 2nd minute obtained an error of 14.30% for sanitation and in the 2nd minute, an error of 4.43% was obtained for the disinfectant limit. While the maximum limit, in the 2nd minute an error of 11.68% is obtained for sanitation, and in the 10th minute an error of 11.64% is obtained for the disinfectant. Likewise, for the results of Fuzzy Logic calculations that have been obtained for sanitation, the output value is 10.20% for Sanitary Alcohol, 10.78% for Glycerin, and 12.62% for H2O2. Likewise the results of the disinfectant output value of 12.56% for Alcohol Disinfectant, 15.58% for Bayclin, and 10.61% for Wipol.

***Keywords**: Fuzzy Logic Controller, loadcell, and DC water pump*

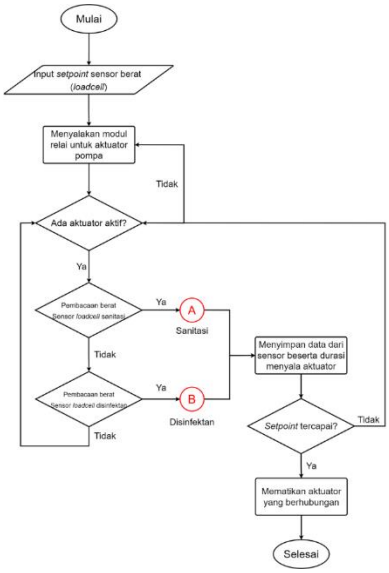
1. Pendahuluan

Dalam upaya untuk membuat pembangunan berkelanjutan di tengah pandemi COVID-19 dibuatlah sebuah Alat & Dispenser Desinfektan dan *Hand Sanitizer* otomatis dengan menanamkan teknologi *touchless* menggunakan *Fuzzy Logic Controller* metode Sugeno sebagai pengampil keputusan. Pada penilitian ini bertujuan untuk mendukung kebiasaan tanpa setuhan. Alat Tugas Akhir ini menggunakan pompa air DC sebagai aktuator yang mengalirkan cairan – cairan dari titik satu ke titik lainnya yang sudah disesuaikan oleh peneliti. Sensor yang digunakan untuk menimbang tangki penampung adalah sensor berat (*loadcell*). Parameter yang akan diambil adalah parameter batas minimum dan maksimum untuk sensor *loadcell*, kemudian hasil *output* dari kontroler *Fuzzy Logic*. Parameter dalam *fuzzy logic* ini adalah hanya untuk melihat *error* dari *output* yang dihasilkan cairan – cairan bahan pembuatan sanitasi dan disinfektan bukan untuk mengakuratkan hasil keluaran dalam pengguna/*user*. Untuk batasan minimum dan maksimum, penulis sudah menyesuaikan dalam laporan[1].

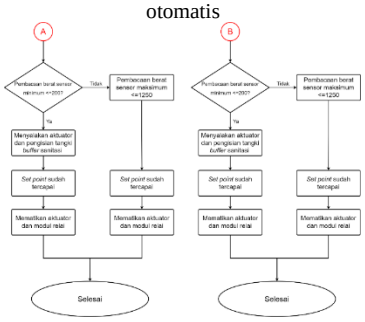
2. Metode

2.2 Perancangan Algoritma dan *Flowchart* Alat Subsystem Prototype Dispenser Otomatis

Berikut merupakan *flowchart* alat secara keseluruhan subsistem prototipe dispenser otomatis yang ditunjukkan pada Gambar 2.1 :



Gambar 2. 15 Perancangan Algoritma dan *Flowchart* Alat Subsistem Prototipe dispenser otomatis



Gambar 2. 16 *Flowchart* lanjutan

Penjelasan *flowchart* alat secara keseluruhan subsistem prototipe dispenser otomatis pada Gambar 2.1 adalah sebagai berikut :

1. Dimulai dengan memasukkan *setpoint* yang sudah diatur ke dalam sensor *loadcell.*
2. Selanjutnya menyalakan modul relai pada aplikasi Arduino untuk mengaktifkan aktuator pompa.
3. Selanjutnya meneruskan sistem apakah ada aktuator yang aktif atau belum, jika belum ada aktuator yang aktif maka akan kembali untuk menyalakan modul relai yang akan mencoba untuk mengaktifkan aktuatornya.
4. Selanjutnya apabila sistem perancangan alat dispenser otomatis ini sudah dapat berjalan, maka dapat dilakukan pembacaan berat pada

sensor *load cell*, pada pembacaan ini dibagi menjadi dua bagian untuk pembacaan pada proses sanitasi dan disinfektan, yang akan dijelaskan sesuai dengan Gambar 2.2.

5. Setelah pembacaan sensor *load cell* ini sudah berhasil kemudian data dari sensor berat ini akan tersimpan pada EEPROM yang akan mengatur durasi nyalanya aktuator.
6. Selanjutnya akan masuk ke sistem apakah sudah tercapai sesuai dengan *set point* yang sudah ditentukan oleh peneliti atau belum. Jika belum, maka proses akan kembali menuju opsi apakah aktuator sudah aktif.
7. Selanjutnya setelah sistem sudah mencapai *set point* yang ditentukan, mematikan aktuator yang berhubungan pada sanitasi dan disinfektan.
8. Selesai

2.3 Perancangan Perangkat Keras (*Hardware*)

Pada perancangan perangkat keras (*hardware*) dan diagram blok ini terdiri dari perancangan diagram blok subsistem prototipe dispenser otomatis menggunakan *Fuzzy Logic* Sugeno dari modul mikrokontroler yang terhubung dengan beberapa aktuator pendukung seperti pompa air DC dan sensor *loadcell*[2][3].

2.4 Rancang Diagram Blok Subsistem Prototipe Dispenser Otomatis

Berikut merupakan rancang diagram blok subsistem prototipe dispenser otomatis yang ditunjukkan pada Gambar 2.3 :

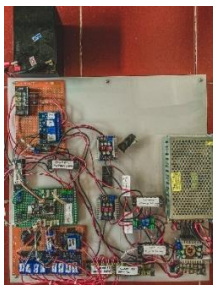
Gambar 2. 17 diagram blok sistem proses pengisian tangki buffer

Penjelasan rancang diagram blok subsistem prototipe PLTMH pada Gambar 2.3 adalah sebagai berikut :

Pada rancang diagram blok subsistem prototipe dispenser otomatis ini menggunakan model *closed loop* yang terdapat *feedback* saat mencapai *set point* yang sudah ditentukan oleh penulis. Dalam penelitian ini, penulis menggunakan pompa air DC sebagai aktuatornya dan sensor untuk *feedback* menggunakan sensor berat yaitu sensor *loadcell*. Sensor *loadcell* ini memiliki fungsi sebagai dua pembatas yaitu batas minimum (<=200 gram) dan batas maksimum (<=900 gram), jadi ketika cairan terbaca dibawah batas minimum maka mengisi dari tabung bahan menuju tabung penampung begitu sensor *loadcell* sudah membaca menyentuh batas maksimum maka aktuator akan berhenti bekerja dan cairan pun akan berhenti mengalir pula.

2.5 Perangkat Keras Alat Secara Keseluruhan

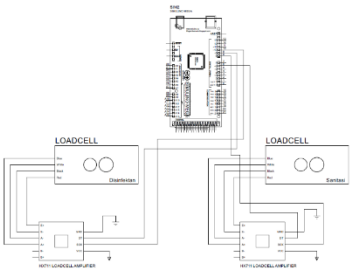
Berikut merupakan perangkat keras alat secara keseluruhan subsistem prototipe dispenser otomatis yang ditunjukkan pada Gambar 2.4 :



Gambar 2. 18 Perangkat Keras Alat Secara Keseluruhan

2.6 Perangkat Keras Sensor *Loadcell*

Berikut merupakan perangkat keras sensor *loadcell* subsistem prototipe dispenser otomatis yang ditunjukkan pada Gambar 2.5 :



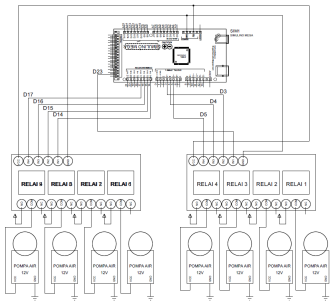
Gambar 2. 19 Perangkat Keras sensor loadcell

Pada sensor *load cell* terdapat 4 pin warna *blue*, *white*, *black*, dan *red* yang terhubung dengan kaki modul HX711 sebagai penguat sensor tersebut E+ (*blue*), E- (*white*), A- (*black*), dan A+ (*red*). Kemudian VCC dan GND terhubung pada *ground* dan sumber untuk kedua sensor *load cell*.

Perancangan alat dispenser otomatis ini menggunakan sensor *load cell* yang dihubungkan pada PORT DIGITAL 6 (DT) dan 8 (SCK) untuk *load cell* hand sanitasi, sedangkan sensor *load cell* untuk disinfektan dihubungkan pada PORT DIGITAL 9 (DT) dan 10 (SCK).

2.7 Perangkat Keras Mikrokontroler dengan Relai 12V

Berikut merupakan perangkat keras mikrokontroler ATMega2560 Promini dengan relai 12V subsistem prototipe dispenser otomatis yang ditunjukkan pada Gambar 2.6 :



Gambar 2. 20 Perangkat keras mikrokontroler, relai dan Pompa air DC

Pada perancangan perangkat keras ini subsistem prototipe dispenser otomatis, merupakan rangkaian relai interkoneksi dengan mikrokontroler Arduino ATMega 2560 Promini perancangan alat dispenser otomatis. Modul relai ini bekerja dengan mendapatkan *input* PORT DIGITAL dari Arduino ATMega yang dimana pin COM pada relai terhubung dengan aktuator (pompa air DC 12v). Untuk alokasi PORT DIGITAL relai terhubung pada pin 23 (relai 1), pin 3 (relai 2), pin 4 (relai 3), pin 5 (relai 4), pin 14 (relai 6), pin 15 (relai 7), pin 16 (relai 8), dan pin 17 (relai 9), sesuai dengan gambar berikut.

3. Hasil dan Analisis

3.1 Pengujian Pada Pengukuran Pembacaan Sensor *Loadcell*

Pengujian untuk pengukuran sensor berat ini dilakukan dengan adanya batasan minimum dan batasan maksimum untuk mengaktifkan aktuator bekerja aktif menggunakan relai. Pengujian ini dilakukan setiap 2 menit sekali sampai pada menit ke 10 yang terbaca pada sensor dan terdapat sebuah *error* yang berbeda setiap menitnya dibandingkan dengan tangki *buffer* yang menggunakan gelar takar seberat 1 liter.

Error dapat dilihat seperti pada tabel 3.1:

Tabel 3. 5. Pengukuran Berat Batas Minimum Sensor *Loadcell*

No.	Pembacaan Sensor Batas Minimum (gram)		Gelas Ukur (gram)		Error (%)		Pembacaan saat menit ke -
	Tangki Sanitasi	Tangki Disinfektan	Tangki Sanitasi	Tangki Disinfektan	Tangki Sanitasi	Tangki Disinfektan	
1.	464,30	374,43	450	370	14,30	4,43	Menit ke 2
2.	464,12	374,30	450	370	14,12	4,30	Menit ke 4
3.	463,92	374,24	450	370	13,92	4,24	Menit ke 6
4.	463,65	373,81	450	370	13,65	3,81	Menit ke 8
5.	463,49	374,32	450	370	13,49	4,32	Menit ke 10

Pada Tabel 3.1 dilakukan pengujian batas minimum pembacaan sensor *loadcell* selama 10 menit dan diamati ketika 2 menit sekali untuk menentukan adanya *error* pada pengujian. Bisa dilihat pada tabel diatas terdapat *error* yang cukup besar untuk sanitasi sebesar 14,30% saat pembacaan menit pertama, sedangkan pada pembacaan menit pertama *error* tangki disinfektan kurang dari 5% yaitu 4,43%. Begitupula untuk menit seterusnya selama 2 menit sekali, pada pembacaan *error* tangki sanitasi terbaca 14,12%, 13,92%, 13,65%, dan 13,49%. Sedangkan untuk setiap 2 menit berikutnya *error* tangki disinfektan terbaca 4,30%, 4,24%, 3,82%, dan 4,32%.

Tabel 3. 6. Pengukuran Berat Batas Maksimum Sensor *Loadcell*

No.	Pembacaan Sensor Batas Maksimum (gram)		Gelas Ukur (gram)		Error		Pembacaan saat menit ke -
	Tangki Sanitasi	Tangki Disinfektan	Tangki Sanitasi	Tangki Disinfektan	Tangki Sanitasi	Tangki Disinfektan	
1.	941,68	961,48	930	950	11,68	11,48	Menit ke 2
2.	941,61	961,50	930	950	11,61	11,50	Menit ke 4
3.	941,54	961,68	930	950	11,54	11,68	Menit ke 6
4.	941,44	961,32	930	950	11,44	11,32	Menit ke 8
5.	941,30	961,64	930	950	11,30	11,64	Menit ke 10

Pada Tabel 3.2 dilakukan pengujian batas maksimum pembacaan sensor *loadcell* selama 10 menit dan diamati ketika 2 menit sekali untuk menentukan adanya *error* pada pengujian. Bisa dilihat pada tabel diatas terdapat *error* yang cukup besar untuk sanitasi sebesar 11,68% saat pembacaan menit pertama, sedangkan pada pembacaan menit pertama *error* tangki disinfektan memiliki pembacaan yang tidak terlalu signifikan jauh yaitu 11,48%. Begitupula untuk menit seterusnya selama 2 menit sekali, pada pembacaan *error* tangki sanitasi terbaca 11,61%, 11,54%, 11,44%, dan 11,30%. Sedangkan untuk setiap 2 menit berikutnya *error* tangki disinfektan terbaca 11,50%, 11,68%, 11,32%, dan 11,64%.

3.2 Pengujian *Fuzzy Logic Control* pada *Volume* Cairan

Logika *Fuzzy Logic* digunakan untuk mengendalikan volume berdasarkan komposisi masing – masing tabung bahan yaitu *alkohol sanitasi, glycerin, H_2O_2, alkohol disinfektan, wipol, dan bayclin.* Variabel yang digunakan untuk logika *Fuzzy* adalah nilai *error* setiap tabung bahan yang didapat dari selisih antara nilai *set point* yang telah ditetapkan oleh peneliti. Dari variabel tersebut akan digunakan untuk menentukan besar persentase volume pada hasil *output*. Pada pengujian logika *Fuzzy Logic* pada Tugas Akhir ini menggunakan metode Sugeno. Pengujian ini dilalukan dengan cara menghitung besar volume yang didapatkan oleh perhitungan logika *Fuzzy Logic* secara manual dengan hasil nilai nilai volume sesuai program. Perhitungan ini digunakan untuk membandingkan hasil volume dalam perhitungan dan volume yang didapatkan oleh sistem.

Tabel 3. 7. Pengujian error sanitasi

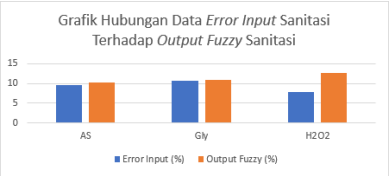
No	Bahan	*Set point* (%)	*Error* (%)
1	Alkohol Sanitasi	15	9,50
2	Glycerin	30	10,65
3	H_2O_2	46	7,88

Berdasarkan data pada Tabel 3.3 dapat diketahui nilai *set point* yang digunakan diambil secara acak. Namun, masih berdasarkan *rule Fuzzy* yang sudah ditentukan oleh peneliti. Nilai *error* didapatkan dari hasil pengujian (tidak terdapat *feedback* karena *open loop*). Pada nilai *error* ini adalah nilai yang absolut, jadi nilai *error* tersebut digunakan sebagai masukan *Fuzzy* untuk menghasilkan volume yang tepat dalam proses pemcampuran sanitasi. Dari tabel diatas didapatkan nilai rata – rata *error* untuk AS (Alkohol Sanitasi) sebanyak 17,50%, *error* Gly (Glycerin) sebanyak 22,65%, dan *error* H_2O_2 sebanyak 24,61%. Data *error* yang besar ini disebabkan karena standar pencampuran yang digunakan oleh peneliti berbeda dengan standar pencampuran sesungguhnya. Hasil pembacaan kontroler *Fuzzy Logic Control* metode Sugeno dapat dilihat pada Tabel 3.4.

Tabel 3. 8. Hasil pengujian kontroler sanitasi

No	Bahan	*Error input* (%)	*Output Fuzzy* (%)	Volume (ml)
1	AS	9,50	10,20	12,80
2	Gly	10,65	10,78	15,23
3	H_2O_2	7,88	12,61	12,59

Berdasarkan data pada tabel 4.4 dapat diketahui rata – rata nilai keluaran *Fuzzy* untuk Alkohol Sanitasi (AS) sebesar 10,20%, Glycerin (Gly) sebesar 10,78%, dan H_2O_2 sebesar 12,61%. Sedangkan untuk rata- rata volume Alkohol Sanitasi (AS) sebesar 12,80 ml, Glycerin (Gly) sebesar 15,23 ml, dan H_2O_2 sebesar 12,59 ml.



Gambar 3. 24 Grafik hubungan data error input sanitasi terhadap output Fuzzy sanitasi

Berdasarkan pada Gambar 4.1 dapat diketahui bahwa semakin besar *error input* maka presentase volume yang dihasilkan akan semakin besar pula. Maka dari hal tersebut dapat disimpulkan bahwa semakin besar nilai *output Fuzzy* maka volume yang dihasilkan akan semakin besar, dan semakin

kecil nilai *output Fuzzy* maka volume yang dihasilkan juga semakin kecil.

Tabel 3. 9. Hasil error disinfektan

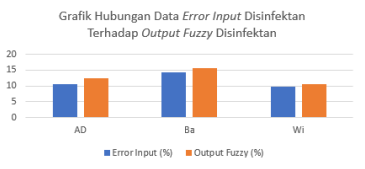
No	Bahan	*Set point* (%)	*Error* (%)
1	Alkohol Disinfektan	20	10,50
2	Bayclin	50	14,23
3	Wipol	35	9,76

Berdasarkan data pada Tabel 3.5 dapat diketahui nilai *set point* yang digunakan diambil secara acak. Namun, masih berdasarkan *rule Fuzzy* yang sudah ditentukan oleh peneliti. Nilai *error* didapatkan dari hasil pengujian (tidak terdapat *feedback* karena sistem *Fuzzy open loop*). Pada nilai *error* ini adalah nilai yang absolut, jadi nilai *error* tersebut digunakan sebagai masukan *Fuzzy* untuk menghasilkan volume yang tepat dalam proses pemcampuran sanitasi. Dari tabel diatas didapatkan nilai rata – rata *error* untuk AD (Alkohol Disinfektan) sebanyak 10,50%, *error* Ba (Bayclin) sebanyak 14,23%, dan *error* Wi (Wipol) sebanyak 9,76%. Data *error* yang besar ini disebabkan karena standar pencampuran yang digunakan oleh peneliti berbeda dengan standar pencampuran sesungguhnya. Hasil pembacaan kontroler *Fuzzy Logic Control* metode Sugeno dapat dilihat pada Tabel 3.6.

Tabel 3. 10. Hasil pengujian kontroler disinfektan

No	Bahan	*Error input* (%)	*Output Fuzzy* (%)	Volume (ml)
1	AD	10,50	12,56	40,76
2	Ba	14,23	15,58	53,35
3	Wi	9,76	10,61	37,89

Berdasarkan data pada tabel 4.2 dapat diketahui rata – rata nilai keluaran *Fuzzy* untuk Alkohol Sanitasi (AS) sebesar 12,56%, Glycerin (Gly) sebesar 15,58%, dan H_2O_2 sebesar 10,61%. Sedangkan untuk rata- rata volume Alkohol Sanitasi (AS) sebesar 40,76 ml, Glycerin (Gly) sebesar 53,35 ml, dan H_2O_2 sebesar 37,89 ml.



Gambar 3. 25 Grafik hubungan data error input disinfektan terhadap output Fuzzy disinfektan

Berdasarkan pada Gambar 4.2 dapat diketahui bahwa semakin besar *error input* maka presentase volume yang dihasilkan akan semakin besar pula. Maka dari hal tersebut dapat disimpulkan bahwa semakin besar nilai *output Fuzzy* maka volume yang dihasilkan akan semakin besar, dan semakin kecil nilai *output Fuzzy* maka volume yang dihasilkan juga semakin kecil.

3.2.1 Perbandingan Perhitungan *Fuzzy* Dengan Hasil Pengujian Pada Alat Dispenser Otomatis

Dilakukan perhitungan *Fuzzy* secara manual untuk membandingkan hasil perhitungan *Fuzzy* dengan hasil pengujian pada alat dispenser otomatis. *Range set point* pada Tugas Akhir ini yaitu sebesar 0 hingga 100%. Perhitungan *Fuzzy* menggunakan nilai yang absolut. Berikut ini perhitungan menggunakan 6 sampel bahan dengan menggunakan logika *Fuzzy*.

1. Perhitungan Pada Alkohol Sanitasi
   Berdasarkan konsep logika *Fuzzy*, terlebih dahulu dihitung nilai *error* dengan perhitungan yang sudah disesuaikan oleh peneliti melalui simulink dan Arduino IDE. Untuk nilai *error input* didapatkan sebesar 9,50 (*Zero*), sedangkan untuk *error Fuzzy* nya terbaca 10,20%.
2. Perhitungan Pada Glycerin
   Berdasarkan konsep logika *Fuzzy*, terlebih dahulu dihitung nilai *error* dengan perhitungan yang sudah disesuaikan oleh peneliti melalui simulink dan Arduino IDE. Untuk nilai *error input* didapatkan sebesar 10,65 (PS), sedangkan untuk *error Fuzzy* nya terbaca 10,78%.
3. Perhitungan Pada H_2O_2
   Berdasarkan konsep logika *Fuzzy*, terlebih dahulu dihitung nilai *error* dengan perhitungan yang sudah disesuaikan oleh peneliti melalui simulink dan Arduino IDE. Untuk nilai *error input* didapatkan sebesar

7,88 (PB), sedangkan untuk *error Fuzzy* nya terbaca 12,61%.

4. Perhitungan Pada Alkohol Disinfektan
   Berdasarkan konsep logika *Fuzzy*, terlebih dahulu dihitung nilai *error* dengan perhitungan yang sudah disesuaikan oleh peneliti melalui simulink dan Arduino IDE. Untuk nilai *error input* didapatkan sebesar 10,50 (*Zero*), sedangkan untuk *error Fuzzy* nya terbaca 12,56%.
5. Perhitungan Pada Bayclin
   Berdasarkan konsep logika *Fuzzy*, terlebih dahulu dihitung nilai *error* dengan perhitungan yang sudah disesuaikan oleh peneliti melalui simulink dan Arduino IDE. Untuk nilai *error input* didapatkan sebesar 14,24 (PS), sedangkan untuk *error Fuzzy* nya terbaca 15,58%.
6. Perhitungan Pada Wipol
   Berdasarkan konsep logika *Fuzzy*, terlebih dahulu dihitung nilai *error* dengan perhitungan yang sudah disesuaikan oleh peneliti melalui simulink dan Arduino IDE. Untuk nilai *error input* didapatkan sebesar 9,76 (PB), sedangkan untuk *error Fuzzy* nya terbaca 10,61%.

3.3 Pengujian Alat Dispenser Otomatis Secara Keseluruhan

Pada pengujian alat keseluruhan perancangan dispenser otomatis akan diujikan terhadap hasil akhir dari keluaran sistem. Terdapat masing – masing 3 keluaran untuk cairan dalam besaran mililiter, pengujian ini akan berfokus pada pengujian proses pengisian sanitasi maupun desinfektan terhadap *user* atau pengguna dimana akan menerima hasil yang sesuai dengan aplikasi sebanyak 100ml, 150ml, dan 200ml.

A. Pengujian saat 100ml pada kedua cairan.
   Pada pengujian kali ini, telah dilakukan secara keseluruhan untuk memproses kedua cairan yaitu sanitasi maupun desinfektan sebanyak 100ml dalam sekali percobaan. Gambar 3.3 dan Gambar 3.4 merupakan bukti bahwa pengujian untuk 100ml cairan sanitasi maupun desinfektan telah berhasil.

Gambar 3. 26 Pengujian 100ml cairan sanitasi

Pada gambar diatas merupakan pengujian cairan sanitasi saat keluaran dalam sistem sebanyak 100ml pengujian kedua pada Tabel 3.7.

Tabel 3. 11 Pengujian 100ml cairan sanitasi

Pengujian ke -	Pengujian yang terukur menggunakan gelas ukur (ml)	*Set point* (%)	*Error* (%)
1	90	100	10
2	100	100	0
3	120	100	20
4	100	100	0

Pada Tabel 3.7 menunjukkan bahwa penguji melakukan pengujian keseluruhan sebanyak empat kali, dan pada pengujian ke – 2 terlihat bahwa tidak terdapatnya *error* dikarenakan keluaran aliran cairan yang tidak selalu sama

Gambar 3. 27 Pengujian 100ml cairan desinfektan

Pada gambar diatas merupakan pengujian cairan desinfektan saat keluaran dalam sistem sebanyak 100ml untuk pengujian ketiga dan keempat pada Tabel 3.8.

Tabel 3. 12 Pengujian 100ml cairan desinfektan

Pengujian ke -	Pengujian yang terukur menggunakan gelas ukur (ml)	*Set point* (%)	*Error* (%)
1	90	100	10
2	110	100	10
3	100	100	0
4	100	100	0

Pada Tabel 3.8 menunjukkan bahwa penguji melakukan pengujian keseluruhan sebanyak empat kali, dan pada pengujian ketiga dan keempat terlihat bahwa tidak terdapatnya *error* dikarenakan keluaran aliran cairan yang tidak selalu sama.

B. Pengujian saat 150ml pada kedua cairan.

Pada pengujian kali ini, telah dilakukan secara keseluruhan untuk memproses kedua cairan yaitu sanitasi maupun desinfektan sebanyak 150ml dalam sekali percobaan. Gambar 3.5 dan Gambar 3.6 merupakan bukti bahwa pengujian untuk 150ml cairan sanitasi maupun desinfektan telah berhasil.

Gambar 3.11. Pengujian 150ml cairan sanitasi

Pada gambar diatas merupakan pengujian cairan sanitasi saat keluaran dalam sistem sebanyak 170ml pada pengujian kedua dalam Tabel 3.9.

Tabel 3.13. Pengujian 150ml cairan sanitasi

Pengujian ke -	Pengujian yang terukur menggunakan gelas ukur (ml)	*Set point* (%)	*Error* (%)
1	160	150	10
2	170	150	20
3	180	150	30
4	150	150	0

Pada Tabel 3.9 menunjukkan bahwa penguji melakukan pengujian keseluruhan sebanyak empat kali, dan pada pengujian ke – 4 terlihat bahwa tidak terdapatnya *error* dikarenakan keluaran aliran cairan yang tidak selalu sama.

Gambar 3.12. Pengujian 150ml cairan desinfektan

Pada gambar diatas merupakan pengujian cairan desinfektan saat keluaran dalam sistem sebanyak 150ml pada pengujian ketiga dalam Tabel 3.10.

Tabel 3.14. Pengujian 150ml cairan desinfektan

Pengujian ke -	Pengujian yang terukur menggunakan gelas ukur (ml)	*Set point* (%)	*Error* (%)
1	140	150	10
2	160	150	10
3	150	150	0
4	190	150	40

Pada Tabel 3.10 menunjukkan bahwa penguji melakukan pengujian keseluruhan sebanyak empat kali, dan pada pengujian ke – 3 terlihat bahwa tidak terdapatnya *error* dikarenakan aliran cairan yang tidak selalu sama untuk keluarannya.

C. Pengujian saat 200ml pada kedua cairan.

Pada pengujian kali ini, telah dilakukan secara keseluruhan untuk memproses kedua cairan yaitu sanitasi maupun desinfektan sebanyak 200ml dalam sekali percobaan. Gambar 3.7 dan Gambar 3.8 merupakan bukti bahwa pengujian untuk 200ml cairan sanitasi maupun desinfektan telah berhasil.

Gambar 3.13. Pengujian 200ml cairan sanitasi

Pada gambar diatas merupakan pengujian cairan sanitasi saat keluaran dalam sistem sebanyak 220ml.

Tabel 3.15. Pengujian 200ml cairan sanitasi

Pengujian ke -	Pengujian yang terukur menggunakan gelas ukur (ml)	*Set point* (%)	*Error* (%)
1	220	200	20
2	230	200	30
3	200	200	0
4	210	200	10

Pada Tabel 3.11 menunjukkan bahwa penguji melakukan pengujian keseluruhan sebanyak empat kali, dan pada pengujian ke – 3 terlihat bahwa tidak terdapatnya *error* dikarenakan aliran cairan yang tidak selalu sama untuk keluarannya.

Gambar 3.14. Pengujian 200ml cairan desinfektan

Pada gambar diatas merupakan pengujian cairan desinfektan saat keluaran dalam sistem sebanyak 200ml.

Tabel 3.16. Pengujian 200ml cairan desinfektan

Pengujian ke -	Pengujian yang terukur menggunakan gelas ukur (ml)	*Set point* (%)	*Error* (%)
1	200	200	0
2	210	200	10
3	220	200	20
4	200	200	0

Pada Tabel 3.12 menunjukkan bahwa penguji melakukan pengujian keseluruhan sebanyak empat kali, dan pada pengujian pertama dan keempat terlihat bahwa tidak terdapatnya *error* dikarenakan aliran cairan yang tidak selalu sama untuk keluarannya.

4. Penutup

4.1 Kesimpulan

Berdasarkan hasil pengujian dan analisis yang telah dilakukan, maka dapat diambil kesimpulan sebagai berikut :

1. Telah berhasilnya dirancang alat pengisian dispenser otomatis dengan terdapatnya *error* 10% dan 20% pada saat proses pengujian cairan sanitasi untuk 100ml dalam tabung ukur. Terdapat *error* yang paling tinggi sebesar 40% pada saat proses pengujian cairan desinfektan 150ml pada pengujian keempat.
2. Untuk hasil pembacaan *error Fuzzy Logic* bahan sanitasi sebesar 10,20% untuk alkohol, 10,78% untuk glycerin, dan H_2O_2 sebesar 12,61% dan didapatkan juga volume sebanyak 12,80 ml untuk alkohol sanitasi, 15,23 ml untuk glycerin, dan 12,59 ml untuk H_2O_2.
3. Untuk hasil pembacaan *error Fuzzy Logic* bahan sanitasi sebesar 12,56% untuk alkohol, 15,58% untuk bayclin, dan wipol sebesar 10,61% dan didapatkan juga volume sebanyak 40,76 ml untuk alkohol disinfektan, 53,35 ml untuk bayclin, dan 37,89 ml untuk wipol.
4. Telah berhasil dirancang protoripe alat dispenser otomatis dengan sistem *Fuzzy Logic Control* yang menghasilkan cairan sanitasi dan disinfektan sebanyak 100ml, 150ml, dan 200ml.

4.2 Saran

Saran untuk pengembangan pada penelitian selanjutnya, yaitu :

1. Lebih dipertimbangkan lagi untuk keakuratan keluaran pada setiap cairan dimulai dari tabung bahan, tabung *buffer*, sampai dengan ke pengguna / *user*.
2. Mengembangkan metode *Fuzzy Logic Control* dengan metode yang sama atau berbeda yang dimana dapat menambahkan jumlah masukan, sehingga diharapkan sistem dapat bekerja dengan lebih akurat atau presisi.
3. Dapat diteliti lebih lanjut mengenai bahan – bahan olahan dasar untuk sanitasi dan disinfektan.
4. Menggunakan batas maksimum atau minimum yang lebih besar maupun kecil, dan menggunakan sensor berat yang sama atau berbeda.

Referensi

[1] G. T. Tulak, S. Ramadhan, and A. Musrifah, "Edukasi Perilaku Cuci Tangan Pakai Sabun Pada Siswa Untuk Pencegahan Transmisi Penyakit," *JMM (Jurnal Masy. Mandiri)*, vol. 4, no. 1, p. 37, 2020, doi: 10.31764/jmm.v4i1.1702.

[2] Robotshop, "Datasheet 3133-Micro Load Cell (0-5kg)-CZL635," *Robotshop*, p. 4, 2011, [Online]. Available: https://www.robotshop.com/media/files/pdf/datasheet-3133.pdf.

[3] B. Wismarizqa, D. Ana, and R. Wati, "DESAIN SISTEM KENDALI FEEDBACK PLUS FEEDFORWARD PADA STIRRED-TANK HEATING PROCESS BERBASIS LABVIEW," pp. 26–32, 2016.

Biodata

Yohanes Juan Kurniadi (21060118140124) lahir di Tangerang, 14 Juni 1999. Telah menempuh pendidikan mulai dari SD Sugiopranoto SJ (2008 – 2014), SMP Santa Maria 2 (2014 – 2016), dan SMA Thomas Aquino (2016 – 2018). Saat ini penulis sedang menyelesaikan pendidikan di Jurusan S1 Teknik Elektro, konsentrasi Teknik Kontrol Otomat angkatan 2018, Fakultas Teknik, Universitas Diponegoro.

Saya menyatakan bahwa segala informasi yang tersedia di makalah ini adalah benar merupakan hasil karya sendiri, bebas dari plagiat, dan semua karya orang lain telah dikutip dengan benar.

Yohanes Juan Kurniadi
NIM. 21060118140124

Pengesahan

Telah disetujui untuk diajukan pada Sidang Akhir
Semarang, 14 September 2022

Pembimbing 1	Pembimbing 2
Ajub Ajulian Zahra, S.T., M.T.	Sumardi, S.T., M.T.
NIP.197107191998022001	NIP. 196811111994121001

LAMPIRAN B

SENARAI PROGRAM

```
#include <HX711_ADC.h>
#if defined(ESP8266)|| defined(ESP32) || defined(AVR)
#include <EEPROM.h>
#endif

//pins:
const int HX711_dout_1 = 6; //mcu > HX711 no 1 dout pin
const int HX711_sck_1 = 8; //mcu > HX711 no 1 sck pin
const int HX711_dout_2 = 9; //mcu > HX711 no 2 dout pin
const int HX711_sck_2 = 10; //mcu > HX711 no 2 sck pin

//HX711 constructor (dout pin, sck pin)
HX711_ADC LoadCell_1(HX711_dout_1, HX711_sck_1); //HX711 1
HX711_ADC LoadCell_2(HX711_dout_2, HX711_sck_2); //HX711 2

const int calVal_eepromAdress_1 = 0; // eeprom adress for
calibration value load cell 1 (4 bytes)
const int calVal_eepromAdress_2 = 4; // eeprom adress for
calibration value load cell 2 (4 bytes)
unsigned long t = 0;

#define relay1 23             //digital pin relay bahan Bayclin//
#define relay2 3              //digital pin relay bahan Wipol//
#define relay3 4              //digital pin relay bahan Dettol//
#define relay4 5                //digital pin relay bahan Alkohol
Desinfektan//
#define relay5 7              //digital pin relay bahan H202//
#define relay6 14             //digital pin relay bahan Glycerin//
#define relay7 15               //digital pin relay bahan Alkohol
Sanitasi//
#define relay8 16            //digital pin relay Buffer Sanitasi//
#define relay9 17          //digital pin relay Buffer Desinfektan//

int kanan = 30, kiri = 31;    //motor dc2 Desinfektan,Sanitasi//
```

```
void setup(){
  Serial.begin(57600); delay(5000);
  Serial.println();
  Serial.println("Starting...");

  float calibrationValue_1; // calibration value load cell 1
  float calibrationValue_2; // calibration value load cell 2

  calibrationValue_1 = 696.0; // uncomment this if you want to set
this value in the sketch
  calibrationValue_2 = 733.0; // uncomment this if you want to set
this value in the sketch
#if defined(ESP8266) || defined(ESP32)
  //EEPROM.begin(512); // uncomment this if you use ESP8266 and
want to fetch the value from eeprom
#endif
  EEPROM.get(calVal_eepromAdress_1,     calibrationValue_1);     //
uncomment this if you want to fetch the value from eeprom
  EEPROM.get(calVal_eepromAdress_2,     calibrationValue_2);     //
uncomment this if you want to fetch the value from eeprom

  LoadCell_1.begin();
  LoadCell_2.begin();
  //LoadCell_1.setReverseOutput();
  //LoadCell_2.setReverseOutput();
  unsigned long stabilizingtime = 2000; // tare preciscion can be
improved by adding a few seconds of stabilizing time
  boolean _tare = true; //set this to false if you don't want tare
to be performed in the next step
  byte loadcell_1_rdy = 0;
  byte loadcell_2_rdy = 0;
  while ((loadcell_1_rdy + loadcell_2_rdy) < 2) { //run startup,
stabilization and tare, both modules simultaniously
    if          (!loadcell_1_rdy)          loadcell_1_rdy         =
LoadCell_1.startMultiple(stabilizingtime, _tare);
    if          (!loadcell_2_rdy)          loadcell_2_rdy         =
LoadCell_2.startMultiple(stabilizingtime, _tare);
```

```
  }
  if (LoadCell_1.getTareTimeoutFlag()) {
    Serial.println("Timeout, check MCU>HX711 no.1 wiring and pin designations");
  }
  if (LoadCell_2.getTareTimeoutFlag()) {
    Serial.println("Timeout, check MCU>HX711 no.2 wiring and pin designations");
  }
  LoadCell_1.setCalFactor(calibrationValue_1);    //    user    set calibration value (float)
  LoadCell_2.setCalFactor(calibrationValue_2);    //    user    set calibration value (float)
  Serial.println("Startup is complete");


  pinMode(relay1, OUTPUT);
  pinMode(relay2, OUTPUT);
  pinMode(relay3, OUTPUT);
  pinMode(relay4, OUTPUT);
  pinMode(relay5, OUTPUT);
  pinMode(relay6, OUTPUT);
  pinMode(relay7, OUTPUT);
  pinMode(relay8, OUTPUT);
  pinMode(relay9, OUTPUT);
  pinMode(kanan,  OUTPUT);
  pinMode(kiri,   OUTPUT);


  digitalWrite(relay1, HIGH);
  digitalWrite(relay2, HIGH);
  digitalWrite(relay3, HIGH);
  digitalWrite(relay4, HIGH);
  digitalWrite(relay5, HIGH);
  digitalWrite(relay6, HIGH);
  digitalWrite(relay7, HIGH);
  digitalWrite(relay8, HIGH);
```

```
  digitalWrite(relay9, HIGH);
  digitalWrite(kanan,  HIGH);
  digitalWrite(kiri,   HIGH);
}


void TriggerLoadCell () {
  static boolean newDataReady = 0;
  const int serialPrintInterval = 0; //increase value to slow down
serial print activity


  // check for new data/start next conversion:
  if (LoadCell_1.update()) newDataReady = true;
  LoadCell_2.update();


  //get smoothed value from data set
  if ((newDataReady)) {
    if (millis() > t + serialPrintInterval) {
      float a = LoadCell_1.getData();
      float b = LoadCell_2.getData();
      Serial.print("Load_cell 1 output val: ");
      Serial.print(a);
      Serial.print("    Load_cell 2 output val: ");
      Serial.println(b);
      newDataReady = 0;
      t = millis();
    }
  }


  // receive command from serial terminal, send 't' to initiate
tare operation:
  if (Serial.available() > 0) {
    char inByte = Serial.read();
    if (inByte == 't') {
      LoadCell_1.tareNoDelay();
```

```
      LoadCell_2.tareNoDelay();
    }
  }

  //check if last tare operation is complete
  if (LoadCell_1.getTareStatus() == true) {
    Serial.println("Tare load cell 1 complete");
  }
  if (LoadCell_2.getTareStatus() == true) {
    Serial.println("Tare load cell 2 complete");
  }
}

void HandSanitizer() {
  digitalWrite(relay5, LOW);
  digitalWrite(relay6, HIGH);
  digitalWrite(relay7, HIGH);
  delay(1000);
  digitalWrite(relay5, HIGH);
  digitalWrite(relay6, LOW);
  digitalWrite(relay7, HIGH);
  delay(1000);
  digitalWrite(relay5, HIGH);
  digitalWrite(relay6, HIGH);
  digitalWrite(relay7, LOW);
  delay(1000);
  digitalWrite(relay5, HIGH);
  digitalWrite(relay6, HIGH);
  digitalWrite(relay7, HIGH);
  delay(1000);
  digitalWrite(relay5, HIGH);
  digitalWrite(relay6, HIGH);
  digitalWrite(relay7, HIGH);
```

```
  delay(1000);
  digitalWrite(relay5, HIGH);
  digitalWrite(relay6, HIGH);
  digitalWrite(relay7, HIGH);
  delay(1000);
}

void Disinfektan() {
  digitalWrite(relay1, LOW);
  digitalWrite(relay2, HIGH);
  digitalWrite(relay3, HIGH);
  digitalWrite(relay4, HIGH);
  delay(1000);
  digitalWrite(relay1, HIGH);
  digitalWrite(relay2, LOW);
  digitalWrite(relay3, HIGH);
  digitalWrite(relay4, HIGH);
  delay(1000);
  digitalWrite(relay1, HIGH);
  digitalWrite(relay2, HIGH);
  digitalWrite(relay3, LOW);
  digitalWrite(relay4, HIGH);
  delay(1000);
  digitalWrite(relay1, HIGH);
  digitalWrite(relay2, HIGH);
  digitalWrite(relay3, HIGH);
  digitalWrite(relay4, LOW);
  delay(1000);
  digitalWrite(relay1, HIGH);
  digitalWrite(relay2, HIGH);
  digitalWrite(relay3, HIGH);
  digitalWrite(relay4, HIGH);
  delay(1000);
```

```
}

void BufferMotorDC() {
  digitalWrite(kanan, LOW);
  digitalWrite(kiri,  HIGH);
  delay(1000);
  digitalWrite(kanan, HIGH);
  digitalWrite(kiri,  LOW);
  delay(1000);
  digitalWrite(kanan, HIGH);
  digitalWrite(kiri,  HIGH);
  delay(1000);
}

void BufferHandSanitizer() {
  digitalWrite (relay8, LOW);
  digitalWrite (relay9, HIGH);
  delay(1000);
  digitalWrite (relay8, HIGH);
  digitalWrite (relay9, HIGH);
  delay(1000);
}

void BufferDisinfektan() {
  digitalWrite (relay8, HIGH);
  digitalWrite (relay9, LOW);
  delay(1000);
  digitalWrite (relay8, HIGH);
  digitalWrite (relay9, HIGH);
  delay(1000);
}

//=======================TriggerBatasLoadcell=====================
============
```

```
float Trigger(){
      if (a <= 200){
        digitalWrite(LoadRelay, LOW);
        digitalWrite(relay1, LOW);
        digitalWrite(relay4, LOW);
        digitalWrite(relay7, LOW);
      }
      else if (a >= 900){
        digitalWrite(relay1, HIGH);
        digitalWrite(relay4, HIGH);
        digitalWrite(relay7, HIGH);
        digitalWrite(LoadRelay, HIGH);
    }
    if (b <= 200){
      digitalWrite(LoadRelay, LOW);
        digitalWrite(relay2, LOW);
        digitalWrite(relay3, LOW);
        digitalWrite(relay6, LOW);
    }
    else if (b >= 900){
        digitalWrite(relay2, HIGH);
        digitalWrite(relay3, HIGH);
        digitalWrite(relay6, HIGH);
        digitalWrite(LoadRelay, HIGH);
    }
}

void loop(){
  TriggerLoadCell();
  Trigger();
  HandSanitizer();
  Disinfektan();
  BufferMotorDC();
```

```
  BufferHandSanitizer();
  BufferDisinfektan();
}
/////////////// INISIALISASI VARIABEL FUZZY
float ErrorAlkohol;
float ErrorH2O2;
float ErrorGlycerin;
float Alkohol = 10,20;
float Glycerin = 10,78;
float H2O2 = 12,61;

/////////////// MEMBERSHIP FUNCTION INPUT FUZZY
float ErrorAlkoholZ, ErrorAlkoholPS, ErrorAlkoholPB,
ErrorH2O2Z, ErrorH2O2PS, ErrorH2O2PB,
ErrorGlycerinZ, ErrorGlycerinPS, ErrorGlycerinPB;

/////////////// RULE FUZZY
float rule1, rule1a, rule2, rule2a, rule3, rule3a, rule4, rule4a,
rule5, rule5a, rule6, rule6a, rule7, rule7a, rule8, rule8a, rule9,
rule9a, rule10, rule10a, rule11, rule11a, rule12, rule12a, rule13,
rule13a, rule14, rule14a, rule15, rule15a, rule16, rule16a, rule17,
rule17a, rule18, rule18a, rule19, rule19a, rule20, rule20a, rule21,
rule21a, rule22, rule22a, rule23, rule23a, rule24, rule24a, rule25,
rule25a, rule26, rule26a, rule27, rule27a;

/////////////// OUTPUT DEFUZZYFIKASI
float VolumeAlkohol;
float VolumeH2O2;
float VolumeGlycerin;

/////////////// FUZZYFIKASI ERROR ALKOHOL
float FuzzyfikasiErrorAlkohol(){
float spAlkohol = 0;
```

```
ErrorAlkohol = spAlkohol + Alkohol;
Serial.print("Error Alkohol:"); delay(2000);
Serial.println(ErrorAlkohol);

//Alkohol Zero
if(ErrorAlkohol >= 0 && ErrorAlkohol < 50){
ErrorAlkoholZ = (50 - ErrorAlkohol)/50;
}
else if (ErrorAlkohol > 50 ) {
ErrorAlkoholZ= 0;
}
//Alkohol PS
if (ErrorAlkohol <= 0){
ErrorAlkoholPS= 0;
}
else if (ErrorAlkohol > 0 && ErrorAlkohol <= 50){
ErrorAlkoholPS= (ErrorAlkohol-0)/50;
}
else if (ErrorAlkohol > 50 && ErrorAlkohol < 100){
ErrorAlkoholPS= (100-ErrorAlkohol)/50;
}
else if (ErrorAlkohol >= 100){
ErrorAlkoholPS= 0;
}

//Alkohol PB
if (ErrorAlkohol <= 50){
ErrorAlkoholPB= 0;
}
else if (ErrorAlkohol > 50 && ErrorAlkohol <= 100){
ErrorAlkoholPB= (ErrorAlkohol-50)/50;
}
else if (ErrorAlkohol > 100){
```

```
ErrorAlkoholPB= 1;
}
}

/////////////// FUZZYFIKASI ERROR H2O2
float FuzzyfikasiErrorH2O2() {
float spH2O2 = 0;
ErrorH2O2 = spH2O2 + H2O2;
Serial.print("Error H2O2:"); delay(2000);
Serial.println(ErrorH2O2);

//H2O2 Zero
if (ErrorH2O2 >= 0 && ErrorH2O2 < 50){
  ErrorH2O2Z= (50 - ErrorH2O2)/50;
}
else if (ErrorH2O2 >= 50){
ErrorH2O2Z= 0;
}

//H2O2 PS
if (ErrorH2O2 <= 0){
ErrorH2O2PS= 0;
}
else if (ErrorH2O2 > 0 && ErrorH2O2 <= 50){
ErrorH2O2PS= (ErrorH2O2-0)/50;
}
else if (ErrorH2O2 > 50 && ErrorH2O2 < 100){
ErrorH2O2PS= (100-ErrorH2O2)/50;
}
else if (ErrorH2O2 >= 100){
ErrorH2O2PS= 0;
}
```

```
//H2O2 PB
if(ErrorH2O2 <= 50){
ErrorH2O2PB= 0;
}
else if (ErrorH2O2 > 50 && ErrorH2O2 <= 100){
ErrorH2O2PB= (ErrorH2O2 - 50)/50;
}
else if (ErrorH2O2 > 100){
ErrorH2O2PB= 1;
}
//Serial.print("Error H2O2 Zero:"); delay(2000);
}

/////////////// FUZZYFIKASI ERROR GLYCERIN
float FuzzyfikasiErrorGlycerin() {
float spGlycerin = 0;
ErrorGlycerin = spGlycerin + Glycerin;
Serial.print("Error Glycerin:"); delay(2000);
Serial.println(ErrorGlycerin);

//Glycerin Zero
if(ErrorGlycerin >= 0 && ErrorGlycerin < 50){
ErrorGlycerinZ= (50 - ErrorGlycerin)/50;
}
else if (ErrorGlycerin >= 50){
ErrorGlycerinZ= 0;
}

//Glycerin PS
if (ErrorGlycerin <= 0){
ErrorGlycerinPS= 0;
}
else if (ErrorGlycerin > 0 && ErrorGlycerin <= 50){
```

```
ErrorGlycerinPS= (ErrorGlycerin-0)/50;
}
else if (ErrorGlycerin > 50 && ErrorGlycerin <= 100){
ErrorGlycerinPS= (100-ErrorGlycerin)/50;
}
else if (ErrorGlycerin > 100){
ErrorGlycerinPS= 0;
}

//Glycerin PB
if (ErrorGlycerin <= 50){
ErrorGlycerinPB= 0;
}
else if (ErrorGlycerin > 50 && ErrorGlycerin <= 100){
ErrorGlycerinPB= (ErrorGlycerin - 50)/50;
}
else if (ErrorGlycerin > 100){
ErrorGlycerinPB= 1;
}
}

/////////////// RULE BASE
float Rule(){
rule1 = min (ErrorAlkoholZ, ErrorH2O2Z);
rule1a = min (rule1, ErrorGlycerinZ);

rule2 = min (ErrorAlkoholZ, ErrorH2O2Z);
rule2a = min (rule2, ErrorGlycerinPS);

rule3 = min (ErrorAlkoholZ, ErrorH2O2Z);
rule3a = min (rule3, ErrorGlycerinPB);

rule4 = min (ErrorAlkoholZ, ErrorH2O2PS);
```

```
rule4a = min (rule4, ErrorGlycerinZ);

rule5 = min (ErrorAlkoholZ, ErrorH2O2PS);
rule5a = min (rule5, ErrorGlycerinPS);

rule6 = min (ErrorAlkoholZ, ErrorH2O2PS);
rule6a = min (rule6, ErrorGlycerinPB);

rule7 = min (ErrorAlkoholZ, ErrorH2O2PB);
rule7a = min (rule7, ErrorGlycerinZ);

rule8 = min (ErrorAlkoholZ, ErrorH2O2PB);
rule8a = min (rule8, ErrorGlycerinPS);

rule9 = min (ErrorAlkoholZ, ErrorH2O2PB);
rule9a = min (rule9, ErrorGlycerinPB);

rule10 = min (ErrorAlkoholPS, ErrorH2O2Z);
rule10a = min (rule10, ErrorGlycerinZ);

rule11 = min (ErrorAlkoholPS, ErrorH2O2Z);
rule11a = min (rule11, ErrorGlycerinPS);

rule12 = min (ErrorAlkoholPS, ErrorH2O2Z);
rule12a = min (rule12, ErrorGlycerinPB);

rule13 = min (ErrorAlkoholPS, ErrorH2O2PS);
rule13a = min (rule13, ErrorGlycerinZ);

rule14 = min (ErrorAlkoholPS, ErrorH2O2PS);
rule14a = min (rule14, ErrorGlycerinPS);

rule15 = min (ErrorAlkoholPS, ErrorH2O2PS);
```

```
rule15a = min (rule15, ErrorGlycerinPB);

rule16 = min (ErrorAlkoholPS, ErrorH2O2PB);
rule16a = min (rule16, ErrorGlycerinZ);

rule17 = min (ErrorAlkoholPS, ErrorH2O2PB);
rule17a = min (rule17, ErrorGlycerinPS);

rule18 = min (ErrorAlkoholPS, ErrorH2O2PB);
rule18a = min (rule18, ErrorGlycerinPB);

rule19 = min (ErrorAlkoholPB, ErrorH2O2Z);
rule19a = min (rule19, ErrorGlycerinZ);

rule20 = min (ErrorAlkoholPB, ErrorH2O2Z);
rule20a = min (rule20, ErrorGlycerinPS);

rule21 = min (ErrorAlkoholPB, ErrorH2O2Z);
rule21a = min (rule21, ErrorGlycerinPB);

rule22 = min (ErrorAlkoholPB, ErrorH2O2PS);
rule22a = min (rule22, ErrorGlycerinZ);

rule23 = min (ErrorAlkoholPB, ErrorH2O2PS);
rule23a = min (rule23, ErrorGlycerinPS);

rule24 = min (ErrorAlkoholPB, ErrorH2O2PS);
rule24a = min (rule24, ErrorGlycerinPB);

rule25 = min (ErrorAlkoholPB, ErrorH2O2PB);
rule25a = min (rule25, ErrorGlycerinZ);

rule26 = min (ErrorAlkoholPB, ErrorH2O2PB);
```

```
rule26a = min (rule26, ErrorGlycerinPS);

rule27 = min (ErrorAlkoholPB, ErrorH2O2PB);
rule27a = min (rule27, ErrorGlycerinPB);
}

float Defuzzyfikasi() {
//OUTPUT Volume Alkohol (dalam satuan %)
float BKA = 0;
float BSA = 25;
float BPA = 50;
//OUTPUT Volume H2O2 (dalam satuan %)
float BKH = 0;
float BSH = 50;
float BPH = 100;
//Output Volume Glycerin(dalam satuan %)
float BKG = 0;
float BSG = 50;
float BPG = 100;

VolumeAlkohol = (rule1a*BKA + rule2a*BKA + rule3a*BKA +
rule4a*BKA + rule5a*BKA + rule6a*BKA + rule7a*BKA +
rule8a*BKA + rule9a*BKA + rule10a*BSA + rule11a*BSA +
rule12a*BSA + rule13a*BSA + rule14a*BSA + rule15a*BSA +
rule16a*BSA + rule17a*BSA + rule18a*BSA + rule19a*BPA +
rule20a*BPA + rule21a*BPA + rule22a*BPA + rule23a*BPA +
rule24a*BPA + rule25a*BPA + rule26a*BPA + rule27a*BPA)/
(rule1a + rule2a + rule3a + rule4a + rule5a + rule6a +
rule7a + rule8a + rule9a + rule10a + rule11a + rule12a +
rule13a + rule14a + rule15a + rule16a + rule17a + rule18a +
rule19a + rule20a + rule21a + rule22a + rule23a + rule24a +
rule25a + rule26a + rule27a);
```

```
VolumeH2O2 = (rule1a*BKH + rule2a*BKH + rule3a*BKH +
rule4a*BSH + rule5a*BSH + rule6a*BSH + rule7a*BPH +
rule8a*BPH + rule9a*BPH + rule10a*BKH + rule11a*BKH +
rule12a*BKH + rule13a*BSH + rule14a*BSH + rule15a*BSH +
rule16a*BPH + rule17a*BPH + rule18a*BPH + rule19a*BKH +
rule20a*BKH + rule21a*BKH + rule22a*BSH + rule23a*BSH +
rule24a*BSH + rule25a*BPH + rule26a*BPH + rule27a*BPH)/
(rule1a + rule2a + rule3a + rule4a + rule5a + rule6a + rule7a
+ rule8a + rule9a + rule10a + rule11a + rule12a + rule13a +
rule14a + rule15a + rule16a + rule17a + rule18a + rule19a +
rule20a + rule21a + rule22a + rule23a + rule24a + rule25a +
rule26a + rule27a);

VolumeGlycerin = (rule1a*BKG + rule2a*BSG + rule3a*BPG +
rule4a*BKG + rule5a*BSG + rule6a*BPG + rule7a*BKG +
rule8a*BSG + rule9a*BPG + rule10a*BKG + rule11a*BSG +
rule12a*BPG + rule13a*BKG + rule14a*BSG + rule15a*BPG +
rule16a*BKG + rule17a*BSG + rule18a*BPG + rule19a*BKG +
rule20a*BSG + rule21a*BPG + rule22a*BKG + rule23a*BSG +
rule24a*BPG + rule25a*BKG + rule26a*BSG + rule27a*BPG)/(
rule1a + rule2a + rule3a + rule4a + rule5a + rule6a + rule7a
+ rule8a + rule9a + rule10a + rule11a + rule12a + rule13a +
rule14a + rule15a + rule16a + rule17a + rule18a + rule19a +
rule20a + rule21a + rule22a + rule23a + rule24a + rule25a +
rule26a + rule27a);
}

void setup() {
Serial.begin(57600);
Serial.print ("Starting Fuzzy Sanitasi...");
}
void loop() {
  FuzzyfikasiErrorAlkohol();
```

```
  FuzzyfikasiErrorH2O2();
  FuzzyfikasiErrorGlycerin();
  Rule();
  Defuzzyfikasi();
}
/////////////// INISIALISASI VARIABEL FUZZY
float ErrorAlkohol_1;
float ErrorBayclin;
float ErrorWipol;
float Alkohol_1 = 12,56;
float Bayclin = 15,58;
float Wipol = 10,61;

/////////////// MEMBERSHIP FUNCTION INPUT FUZZY
float ErrorAlkohol_1Z, ErrorAlkohol_1PS, ErrorAlkohol_1PB,
      ErrorBayclinZ, ErrorBayclinPS, ErrorBayclinPB,
      ErrorWipolZ, ErrorWipolPS, ErrorWipolPB;

/////////////// RULE FUZZY
float  rule1_1,  rule1a_1,  rule2_1,  rule2a_1,  rule3_1,  rule3a_1,
rule4_1, rule4a_1,
rule5_1, rule5a_1, rule6_1, rule6a_1, rule7_1, rule7a_1, rule8_1,
rule8a_1, rule9_1,
rule9a_1,  rule10_1,  rule10a_1,  rule11_1,  rule11a_1,  rule12_1,
rule12a_1, rule13_1,
rule13a_1,  rule14_1,  rule14a_1,  rule15_1,  rule15a_1,  rule16_1,
rule16a_1, rule17_1,
rule17a_1,  rule18_1,  rule18a_1,  rule19_1,  rule19a_1,  rule20_1,
rule20a_1, rule21_1,
rule21a_1,  rule22_1,  rule22a_1,  rule23_1,  rule23a_1,  rule24_1,
rule24a_1, rule25_1,
rule25a_1, rule26_1, rule26a_1, rule27_1, rule27a_1;

/////////////// OUTPUT DEFUZZYFIKASI
float VolumeAlkohol_1;
float VolumeBayclin;
```

```
float VolumeWipol;

/////////////// FUZZYFIKASI ERROR ALKOHOL
void FuzzyfikasiErrorAlkohol_1(){
float spAlkohol_1 = 0;
ErrorAlkohol_1 = spAlkohol_1 + Alkohol_1;
Serial.print("Error Alkohol_1:");
Serial.println(ErrorAlkohol_1);

//Alkohol Zero
if(ErrorAlkohol_1 >= 0 && ErrorAlkohol_1 < 50){
ErrorAlkohol_1Z = (50 - ErrorAlkohol_1)/50;
}
else if (ErrorAlkohol_1 > 50 ) {
ErrorAlkohol_1Z= 0;
}
//Alkohol PS
if (ErrorAlkohol_1 <= 0){
ErrorAlkohol_1PS= 0;
}
else if (ErrorAlkohol_1 > 0 && ErrorAlkohol_1 <= 50){
ErrorAlkohol_1PS= (ErrorAlkohol_1-0)/50;
}
else if (ErrorAlkohol_1 > 50 && ErrorAlkohol_1 < 100){
ErrorAlkohol_1PS= (100-ErrorAlkohol_1)/50;
}
else if (ErrorAlkohol_1 >= 100){
ErrorAlkohol_1PS= 0;
}

//Alkohol PB
if (ErrorAlkohol_1 <= 50){
ErrorAlkohol_1PB= 0;
```

```
}
else if (ErrorAlkohol_1 > 50 && ErrorAlkohol_1 <= 100){
ErrorAlkohol_1PB= (ErrorAlkohol_1-50)/50;
}
else if (ErrorAlkohol_1 > 100){
ErrorAlkohol_1PB= 1;
}
}

/////////////// FUZZYFIKASI ERROR BAYCLIN
void FuzzyfikasiErrorBayclin() {
float spBayclin = 0;
ErrorBayclin = spBayclin + Bayclin;
Serial.print("Error Bayclin:");
Serial.println(ErrorBayclin);

//Bayclin Zero
if(ErrorBayclin >= 0 && ErrorBayclin < 50){
ErrorBayclinZ= (50 - ErrorBayclin)/50;
}
else if (ErrorBayclin >= 50){
ErrorBayclinZ= 0;
}

//Bayclin PS
if (ErrorBayclin <= 0){
ErrorBayclinPS= 0;
}
else if (ErrorBayclin > 0 && ErrorBayclin <= 50){
ErrorBayclinPS= (ErrorBayclin-0)/50;
}
else if (ErrorBayclin > 50 && ErrorBayclin <= 100){
ErrorBayclinPS= (100-ErrorBayclin)/50;
```

```
}
else if (ErrorBayclin > 100){
ErrorBayclinPS= 0;
}

//Bayclin PB
if (ErrorBayclin <= 50){
ErrorBayclinPB= 0;
}
else if (ErrorBayclin > 50 && ErrorBayclin <= 100){
ErrorBayclinPB= (ErrorBayclin - 50)/50;
}
else if (ErrorBayclin > 100){
ErrorBayclinPB= 1;
}
}

/////////////// FUZZYFIKASI ERROR Wipol
void FuzzyfikasiErrorWipol() {
float spWipol = 0;
ErrorWipol = spWipol + Wipol;
Serial.print("Error Wipol:");
Serial.println(ErrorWipol);

//Wipol Zero
if (ErrorWipol >= 0 && ErrorWipol < 50){
  ErrorWipolZ= (50 - ErrorWipol)/50;
}
else if (ErrorWipol >= 50){
ErrorWipolZ= 0;
}

//Wipol PS
```

```
if (ErrorWipol <= 0){
ErrorWipolPS= 0;
}
else if (ErrorWipol > 0 && ErrorWipol <= 50){
ErrorWipolPS= (ErrorWipol-0)/50;
}
else if (ErrorWipol > 50 && ErrorWipol < 100){
ErrorWipolPS= (100-ErrorWipol)/50;
}
else if (ErrorWipol >= 100){
ErrorWipolPS= 0;
}

//Wipol PB
if(ErrorWipol <= 50){
ErrorWipolPB= 0;
}
else if (ErrorWipol > 50 && ErrorWipol <= 100){
ErrorWipolPB= (ErrorWipol - 50)/50;
}
else if (ErrorWipol > 100){
ErrorWipolPB= 1;
}
Serial.print("Error Wipol Zero:");
}

/////////////// RULE BASE
void Rule(){
rule1_1 = min (ErrorAlkohol_1Z, ErrorBayclinZ);
rule1a_1 = min (rule1_1, ErrorWipolZ);

rule2_1 = min (ErrorAlkohol_1Z, ErrorBayclinZ);
rule2a_1 = min (rule2_1, ErrorWipolPS);
```

```
rule3_1 = min (ErrorAlkohol_1Z, ErrorBayclinZ);
rule3a_1 = min (rule3_1, ErrorWipolPB);

rule4_1 = min (ErrorAlkohol_1Z, ErrorBayclinPS);
rule4a_1 = min (rule4_1, ErrorWipolZ);

rule5_1 = min (ErrorAlkohol_1Z, ErrorBayclinPS);
rule5a_1 = min (rule5_1, ErrorWipolPS);

rule6_1 = min (ErrorAlkohol_1Z, ErrorBayclinPS);
rule6a_1 = min (rule6_1, ErrorWipolPB);

rule7_1 = min (ErrorAlkohol_1Z, ErrorBayclinPB);
rule7a_1 = min (rule7_1, ErrorWipolZ);

rule8_1 = min (ErrorAlkohol_1Z, ErrorBayclinPB);
rule8a_1 = min (rule8_1, ErrorWipolPS);

rule9_1 = min (ErrorAlkohol_1Z, ErrorBayclinPB);
rule9a_1 = min (rule9_1, ErrorWipolPB);

rule10_1 = min (ErrorAlkohol_1PS, ErrorBayclinZ);
rule10a_1 = min (rule10_1, ErrorWipolZ);

rule11_1 = min (ErrorAlkohol_1PS, ErrorBayclinZ);
rule11a_1 = min (rule11_1, ErrorWipolPS);

rule12_1 = min (ErrorAlkohol_1PS, ErrorBayclinZ);
rule12a_1 = min (rule12_1, ErrorWipolPB);

rule13_1 = min (ErrorAlkohol_1PS, ErrorBayclinPS);
rule13a_1 = min (rule13_1, ErrorWipolZ);
```

```
rule14_1 = min (ErrorAlkohol_1PS, ErrorBayclinPS);
rule14a_1 = min (rule14_1, ErrorWipolPS);

rule15_1 = min (ErrorAlkohol_1PS, ErrorBayclinPS);
rule15a_1 = min (rule15_1, ErrorWipolPB);

rule16_1 = min (ErrorAlkohol_1PS, ErrorBayclinPB);
rule16a_1 = min (rule16_1, ErrorWipolZ);

rule17_1 = min (ErrorAlkohol_1PS, ErrorBayclinPB);
rule17a_1 = min (rule17_1, ErrorWipolPS);

rule18_1 = min (ErrorAlkohol_1PS, ErrorBayclinPB);
rule18a_1 = min (rule18_1, ErrorWipolPB);

rule19_1 = min (ErrorAlkohol_1PB, ErrorBayclinZ);
rule19a_1 = min (rule19_1, ErrorWipolZ);

rule20_1 = min (ErrorAlkohol_1PB, ErrorBayclinZ);
rule20a_1 = min (rule20_1, ErrorWipolPS);

rule21_1 = min (ErrorAlkohol_1PB, ErrorBayclinZ);
rule21a_1 = min (rule21_1, ErrorWipolPB);

rule22_1 = min (ErrorAlkohol_1PB, ErrorBayclinPS);
rule22a_1 = min (rule22_1, ErrorWipolZ);

rule23_1 = min (ErrorAlkohol_1PB, ErrorBayclinPS);
rule23a_1 = min (rule23_1, ErrorWipolPS);

rule24_1 = min (ErrorAlkohol_1PB, ErrorBayclinPS);
rule24a_1 = min (rule24_1, ErrorWipolPB);
```

```
rule25_1 = min (ErrorAlkohol_1PB, ErrorBayclinPB);
rule25a_1 = min (rule25_1, ErrorWipolZ);

rule26_1 = min (ErrorAlkohol_1PB, ErrorBayclinPB);
rule26a_1 = min (rule26_1, ErrorWipolPS);

rule27_1 = min (ErrorAlkohol_1PB, ErrorBayclinPB);
rule27a_1 = min (rule27_1, ErrorWipolPB);
}

void Defuzzyfikasi() {
Rule();
//OUTPUT Volume Alkohol (dalam satuan %)
float BKA_1 = 0;
float BSA_1 = 25;
float BPA_1 = 50;
//Output Volume Bayclin (dalam satuan %)
float BKB = 0;
float BSB = 50;
float BPB = 100;
//OUTPUT Volume Wipol (dalam satuan %)
float BKW = 0;
float BSW = 50;
float BPW = 100;

VolumeAlkohol_1 = (rule1a_1*BKA_1 + rule2a_1*BKA_1 + rule3a_1*BKA_1
+
rule4a_1*BKA_1 + rule5a_1*BKA_1 + rule6a_1*BKA_1 + rule7a_1*BKA_1
+
rule8a_1*BKA_1 + rule9a_1*BKA_1 + rule10a_1*BSA_1 + rule11a_1*BSA_1
+
rule12a_1*BSA_1    +    rule13a_1*BSA_1    +    rule14a_1*BSA_1    +
rule15a_1*BSA_1 +
```

```
rule16a_1*BSA_1    +    rule17a_1*BSA_1    +    rule18a_1*BSA_1    +
rule19a_1*BPA_1 +

rule20a_1*BPA_1    +    rule21a_1*BPA_1    +    rule22a_1*BPA_1    +
rule23a_1*BPA_1 +

rule24a_1*BPA_1    +    rule25a_1*BPA_1    +    rule26a_1*BPA_1    +
rule27a_1*BPA_1)/

(rule1a_1 + rule2a_1 + rule3a_1 + rule4a_1 + rule5a_1 + rule6a_1 +

rule7a_1 + rule8a_1 + rule9a_1 + rule10a_1 + rule11a_1 + rule12a_1
+

rule13a_1  +  rule14a_1  +  rule15a_1  +  rule16a_1  +  rule17a_1  +
rule18a_1 +

rule19a_1  +  rule20a_1  +  rule21a_1  +  rule22a_1  +  rule23a_1  +
rule24a_1 +

rule25a_1 + rule26a_1 + rule27a_1);


VolumeBayclin = (rule1a_1*BKB + rule2a_1*BKB + rule3a_1*BKB +

rule4a_1*BSB + rule5a_1*BSB + rule6a_1*BSB + rule7a_1*BPB +

rule8a_1*BPB + rule9a_1*BPB + rule10a_1*BKB + rule11a_1*BKB +

rule12a_1*BKB + rule13a_1*BSB + rule14a_1*BSB + rule15a_1*BSB +

rule16a_1*BPB + rule17a_1*BPB + rule18a_1*BPB + rule19a_1*BKB +

rule20a_1*BKB + rule21a_1*BKB + rule22a_1*BSB + rule23a_1*BSB +

rule24a_1*BSB + rule25a_1*BPB + rule26a_1*BPB + rule27a_1*BPB)/

(rule1a_1 + rule2a_1 + rule3a_1 + rule4a_1 + rule5a_1 + rule6a_1 +
rule7a_1 +

rule8a_1 + rule9a_1 + rule10a_1 + rule11a_1 + rule12a_1 + rule13a_1
+

rule14a_1  +  rule15a_1  +  rule16a_1  +  rule17a_1  +  rule18a_1  +
rule19a_1 +

rule20a_1  +  rule21a_1  +  rule22a_1  +  rule23a_1  +  rule24a_1  +
rule25a_1 +

rule26a_1 + rule27a_1);


VolumeWipol = (rule1a_1*BKW + rule2a_1*BSW + rule3a_1*BPW +

rule4a_1*BKW + rule5a_1*BSW + rule6a_1*BPW + rule7a_1*BKW +

rule8a_1*BSW + rule9a_1*BPW + rule10a_1*BKW + rule11a_1*BSW +

rule12a_1*BPW + rule13a_1*BKW + rule14a_1*BSW + rule15a_1*BPW +

rule16a_1*BKW + rule17a_1*BSW + rule18a_1*BPW + rule19a_1*BKW +
```

```
rule20a_1*BSW + rule21a_1*BPW + rule22a_1*BKW + rule23a_1*BSW +
rule24a_1*BPW + rule25a_1*BKW + rule26a_1*BSW + rule27a_1*BPW)/(
rule1a_1 + rule2a_1 + rule3a_1 + rule4a_1 + rule5a_1 + rule6a_1 +
rule7a_1 +
rule8a_1 + rule9a_1 + rule10a_1 + rule11a_1 + rule12a_1 + rule13a_1
+
rule14a_1 + rule15a_1 + rule16a_1 + rule17a_1 + rule18a_1 +
rule19a_1 +
rule20a_1 + rule21a_1 + rule22a_1 + rule23a_1 + rule24a_1 +
rule25a_1 +
rule26a_1 + rule27a_1);
}

void setup() {
Serial.begin(57600);
}
void loop() {
  FuzzyfikasiErrorAlkohol_1();
  FuzzyfikasiErrorWipol();
  FuzzyfikasiErrorBayclin();
  Rule();
  Defuzzyfikasi();
}
```