Jurnal Ilmu Komputer dan Informatika (JIKI)
Vol. 1, No. 1, Juni 2021, Hal. 9-28
https://jiki.jurnal-id.com

DOI: https://doi.org/10.54082/jiki.4
P-ISSN: 2807-6664
E-ISSN: 2807-6591

Robot *Air Hockey* Berbasis Pengolahan Citra Dengan Metode Kendali *Proportional-Integral-Derivative* Dan *Fuzzy Logic Controller*

Rafly Hidayatullah[*1], Irfhando Mahendra[*2], Agung Surya Wibowo[3], Muhammad Ridho Rosa[4]

[1,2,3,4]Teknik Elektro, Fakultas Teknik Elektro, Universitas Telkom, Indonesia
Email: [1]raflyh@student.telkomuniversity.ac.id, [2]irfhandomahendra@student.telkomuniversity.ac.id, [3]agungsw@telkomuniversity.ac.id, [4]mridhorosa@telkomuniversity.ac.id

Abstrak

Penelitian ini berisi perancangan implementasi sistem mekatronika robot *air hockey* yang dapat berkompetisi dengan manusia. Sistem akan berfokus pada pergerakan *paddle* untuk sisi robot *air hockey* dengan metode kendali *Proportional-Integral-Derivative* (PID) pada bidang X dan *fuzzy logic controller* (FLC) pada bidang Y. Kamera digunakan sebagai sensor untuk menangkap citra yang akan diolah menggunakan modul *blob detection* guna mendeteksi posisi *puck* dan *paddle*. Sistem robot akan melakukan *tracking* posisi *puck* dan memprediksi arah geraknya, supaya robot dapat menahan dan memukul kembali *puck.* Sistem pengolahan citra yang dirancang memiliki kecepatan maksimum pengambilan citra sebesar 0,0125 detik per *frame* dengan tingkat akurasi sebesar 93%. Penerapan kendali *Proportional-Derivative* (PD) dan prediktor K = 0,5 pada bidang X, serta kendali FLC pada bidang Y menghasilkan respons gerak robot dengan *error* maksimum sebesar 3,44%. Dengan sistem ini, robot berhasil merespons dengan baik datangnya *puck* dengan kecepatan maksimum 2,04 m/s dan mengembalikannya dengan kecepatan maksimum sebesar 2,27 m/s.

Kata kunci: *blob detection*, *fuzzy logic controller*, pengolahan citra, *proportional-integral-derivative*, robot *air hockey*.

Abstract

This research contains the design of the mechatronic system's implementation for an air hockey robot that can compete with humans. The system will focus on the movement of the paddle for the side of the air hockey robot with the PID control method on the X plane and fuzzy logic controller (FLC) on the Y plane. The camera is used as a sensor to capture the image that will be processed using the blob detection module to detect the puck and paddle positions. The robot system will track the puck's position and predict the direction of its motion so that the robot can hold and hit the puck again. The image processing system designed has a maximum image capture speed of 0.0125 seconds per frame with an accuracy rate of 93%. The application of Proportional-Derivative (PD) control and predictor K = 0.5 in the X plane, and FLC control in the Y plane give a robot motion response with a maximum error of 3.44%. With this system, the robot managed to respond nicely to the arrival of the puck with a maximum speed of 2.04 m/s and return it with a maximum speed of 2.27 m/s.

Keywords: air hockey robot, blob detection, fuzzy logic controller, image processing, proportional-integral-derivative controller.

1. PENDAHULUAN

Dewasa ini, perkembangan teknologi robot sangatlah pesat. Robot adalah sebuah unit baik berupa mekanikal atau fisikal maupun virtual yang memiliki kecerdasan. Pada umumnya, robot berupa rangkaian elektromekanik yang dapat bergerak dan memiliki akal. Robot telah masuk dalam berbagai segi kehidupan manusia, seperti bidang industri, kedokteran, pelayanan, mainan/peliharaan, dan edukasi [1]. Robot yang mampu berkompetisi dengan manusia menjadi hal yang jarang ditemui sekaligus sangat menarik untuk terus digali. Contoh robot yang mampu berkompetisi dengan manusia yang telah diteliti terdapat pada bidang catur. Komputer catur bernama *AlphaZero* tercatat mampu mengalahkan

DOI: https://doi.org/10.54082/jiki.4
P-ISSN: 2807-6664
E-ISSN: 2807-6591

pendahulunya, *Stockfish* dengan rekor 155 menang dan 6 kalah [2]. Pada akhirnya, robot yang bisa bekerja di ruang yang sama dengan manusia menjadi lebih dibutuhkan [3].

Pada konteks ini, robot *air hockey* adalah uji coba yang baik. Ketika robot bermain *air hockey* dengan manusia, robot perlu sesegera mungkin merespons pergerakan bola (*puck*) dengan akurat. Dalam pembuatan robot ini, permasalahan utama terletak pada bagaimana robot memiliki sistem kendali yang responsif dan akurat, juga bagaimana robot mampu mendeteksi *puck*. Jika sistem kendali robot tidak responsif, kemungkinan besar *puck* tidak terpukul tepat waktu (*delay)*. Jika robot tidak akurat, maka robot akan meleset ketika mencoba memukul *puck*. Robot pun perlu memiliki kemampuan untuk membaca pergerakan *puck* dan menentukan titik pergerakan pemukul (*paddle)* agar *puck* meluncur sesuai dengan harapan. Beberapa penelitian sebelumnya terkait robot *air hockey* yaitu "*Autonomous Air Hockey SSYX02-16-82*" oleh Berntsson C. dkk. menghasilkan robot *air hockey* dengan kemampuan mengantisipasi *puck* rata-rata sebesar 3 m/s dan mengembalikannya dengan kecepatan sebesar 1,44 m/s [4], serta "*Autonomous Air Hockey SSYX02-16-11*" oleh Ansgar M. dkk. menghasilkan robot *air hockey* dengan kemampuan mengantisipasi *puck* rata-rata sebesar 7 m/s dan mengembalikannya dengan kecepatan sebesar 1,7 m/s [5].

Dengan alasan tersebut, pada penelitian ini akan dirancang mengenai pengendalian gerak robot untuk menggerakkan *paddle* pada *air hockey*. Metode pengendalian gerak robot akan menggunakan kendali *Proportional-Integral-Derivative* (PID) yang disematkan algoritma prediktor pada bidang X (gerak ke kanan dan kiri) serta *fuzzy logic controller* (FLC) pada bidang Y (gerak maju dan mundur). Untuk mendeteksi pergerakan *puck*, akan digunakan kamera sebagai sensor dan diproses dengan pengolahan citra menggunakan modul *blob detection*. Sistem robot akan melakukan *tracking* terhadap pergerakan *puck* dengan kendali PID dan prediksi arah tujuan *puck* dengan algoritma prediktor supaya *paddle* dapat menahan *puck* lalu dengan kendali FLC robot akan memukul kembali *puck* ke sisi lawan.

2. METODE PENELITIAN

2.1. Permainan Papan *Air Hockey*

Air Hockey merupakan sebuah bentuk permainan papan yang pada umumnya dimainkan oleh dua orang. Masing-masing pemain bertujuan untuk mencetak skor dengan cara memasukkan bola (*puck)* ke gawang lawan. *Puck* berbentuk lingkaran dan pipih, ukurannya tidak memiliki aturan khusus. Berikut contoh bentuk *puck* pada permainan *air hockey*. *Air hockey* dimainkan pada sebuah papan yang memiliki gaya gesek kecil. Papan *air hockey* pada umumnya memiliki lubang-lubang kecil untuk memberikan tekanan udara dari bawah papan agar mengurangi gesekan antara *puck* dan *paddle*. Berdasarkan keputusan *United States Air Hockey Association* (USSA) dan *Air Hockey Player Association* (AHPA), ukuran papan memiliki regulasi untuk sebuah turnamen *air hockey*, yaitu panjangnya 8 kaki 3,5 inci atau setara dengan 243,84 cm dan lebarnya 4 kaki 3.25 inci atau setara dengan 130,175 cm [6].

Tabel 1. Daftar Ukuran Papan *Air Hockey*

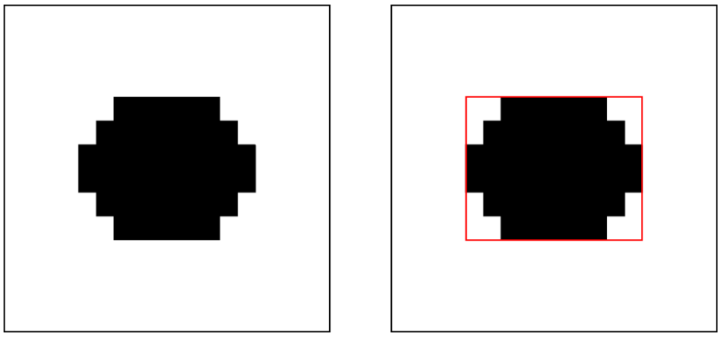
Tipe	Panjang (kaki)	Lebar (kaki)
Full Size Table	6 – 8	4 – 6
Regulation Size Table	8 – 8.5	4 – 6
Ukuran Standar	7 – 8	4 – 5
Arcade Table	6 – 8	4 – 6
Ukuran Rata-rata	5 – 6	3 – 4
Ukuran Anak-anak	4 – 5	2 – 3

DOI: https://doi.org/10.54082/jiki.4
P-ISSN: 2807-6664
E-ISSN: 2807-6591

Pada dasarnya baik ukuran *puck, paddle*, maupun papan permainan dibuat agar tetap proporsional seperti yang ditunjukkan pada Tabel 1. Apabila papan dibuat besar, maka *puck* dan *paddle* pun memiliki ukuran relatif besar apabila dibandingkan dengan *puck* dan *paddle* untuk dimainkan pada papan yang lebih kecil.

2.2. *Blob Detection* Pendeteksi Posisi *Puck* dan *Paddle*

Blob adalah sekumpulan piksel pada suatu gambar yang memiliki sifat yang sama, seperti warna yang sama atau tingkat *grayscale* yang sama. Setiap piksel yang tergabung didaerah *blob* akan berada di bagian depan, sementara piksel-piksel yang berada di bagian belakang akan memiliki nilai logika *zero* dan dianggap sebagai *background*. Dengan begitu, piksel *non-zero* merupakan bagian dari objek yang ingin dideteksi. *Blob* mampu mendeteksi titik-titik piksel yang memiliki kecerahan warna yang mirip dari latar belakang dan menyatukannya ke dalam suatu *region*, dengan begitu *blob* digunakan untuk mengisolasi objek dengan warna khusus dan mengeliminasi latar gambar yang tidak terpakai. Dengan kata lain konsep *blob* di sini adalah pengelompokan satu piksel dengan piksel lain yang hampir serupa menggunakan konsep ketetanggaan dan *labelling*, kemudian memisahkannya menjadi bagian-bagian citra [7].



(a) (b)

Gambar 1. Contoh Pengaplikasian *Blob Detection*: (a) Citra kamera, (b) objek yang terdeteksi

Blob detection memanfaatkan beberapa parameter untuk menjalankan algoritmanya. Parameter yang digunakan pada *blob detection* adalah sebagai berikut,

a. *Theresholding*: Parameter ini akan melakukan konversi gambar sumber menjadi beberapa gambar biner dengan nilai ambang minimum yang telah ditentukan.
b. *Grouping*: Parameter ini akan melakukan pengelompokan pada gambar biner, piksel putih yang terhubung akan dikelompokkan menjadi satu dan disebut dengan *blob* biner
c. *Merging*: Pusat dari *blob* biner pada gambar biner dihitung, lalu *blob* yang letaknya lebih kecil dari jarak minimum antar *blob* akan digabungkan.
d. *Center & Radius Calculation*: Pusat dan jari-jari dari *merged blobs* yang baru dihitung dan dijadikan nilai baru.

Dengan memanfaatkan *blob detection*, *puck* dan *paddle* robot akan dideteksi lokasinya pada sumbu X dan sumbu Y dalam piksel.

2.3. Kendali PID

Untuk mengendalikan pergerakan robot pada bidang X, akan digunakan kendali PID. Penggunaan kendali PID didasari karena kesederhanaannya dalam rancangan, analisa, dan implementasi pada suatu sistem yang memiliki sistem orde satu maupun sistem orde dua, yang pada umumnya seluruh sistem

DOI: https://doi.org/10.54082/jiki.4
P-ISSN: 2807-6664
E-ISSN: 2807-6591

fisik dalam teknik elektro dapat diuraikan menjadi kedua orde sistem tersebut [8]. Kendali PID memiliki tiga buah parameter, yaitu kendali proporsional, kendali integral, dan kendali derivatif, yang bisa diatur agar respons sistem sesuai sebagaimana yang diinginkan. Tabel 2 menunjukkan karakteristik pada setiap kendali PID.

Tabel 2. Tanggapan Sistem Kendali PID terhadap Perubahan Parameter [9]

Parameter Dinaikkan	***Rise Time***	***Overshoot***	***Settling Time***	***Error Steady State***
Proporsional (K_p)	⬇	⬆	Perubahan Kecil	⬇
Integral (K_i)	⬇	⬆	⬆	Mengeliminasi
Derivatif (K_d)	Perubahan Kecil	⬇	⬇	Perubahan Kecil

Terdapat empat macam kendali PID, yaitu kendali P, kendali PI, kendali PD, dan kendali PID. Keempat kendali tersebut digunakan sesuai dengan kebutuhan sistem agar dapat mencapai respons sistem yang diinginkan.

2.4. PID Digital

Kendali PID digital merupakan kendali PID dengan operasi perkalian, integral, dan diferensial dilakukan secara numerik pada komputer digital [10]. Jika kalkulasi numerik dilakukan secara akurat, respons sistem dari kontrol digital akan sangat mirip dengan respons sistem dari kontrol analog. Beberapa metode dapat digunakan untuk mencari ekuivalensi waktu diskrit dari kontroler PID analog. Beberapa diantaranya adalah *backward difference method, forward difference method, bilinear transformation method, impulse-invariance method, step-invariance method* dan *matched pole-zero method* [11]. Metode *backward difference* merupakan metode paling ringkas untuk menentukan integrasi numerik dari suatu sinyal. Metode ini akan menghasilkan filter stabil dalam waktu diskrit ketika diberikan filter stabil dalam waktu kontinu. Dengan menggunakan metode *backward difference*, fungsi transfer integrator numerik ditunjukkan pada persamaan (1).

$$M(z) = \frac{Tz}{z-1} E(z) \tag{1}$$

Fungsi transfer dari diferensiator numerik merupakan kebalikan dari fungsi transfer integrator numerik yang ditunjukkan pada persamaan (1). Dengan begitu, dengan substitusi (1) dan juga kebalikannya ke fungsi transfer kendali PID domain laplace, fungsi transfer kendali PID digital dapat dituliskan menjadi persamaan (2).

$$Gc(z) = K_p + \frac{K_i Tz}{z-1} + \frac{K_D(z-1)}{Tz} \tag{2}$$

Persamaan diferens untuk menghitung siklus (*new duty cycle*) untuk kendali PID digital ditulis dalam persamaan (3).

$$u[k] = K_p e[k] + K_i T \sum_{i=0}^{k} e[i] + \frac{K_D}{T}\{e[k] - e[k-1]\} \tag{3}$$

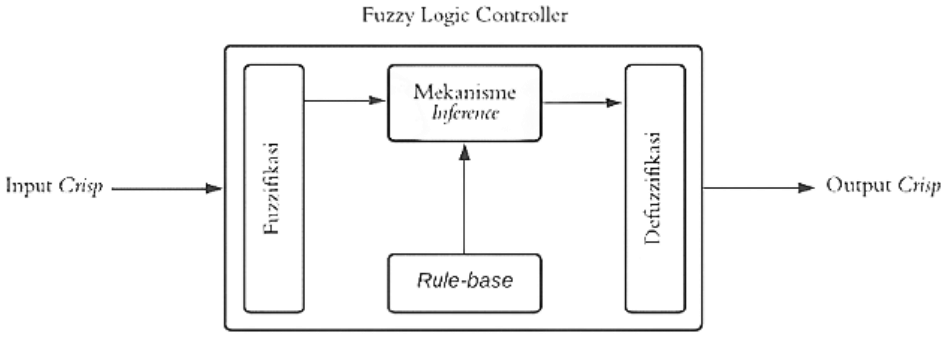
$u[k]$ = *Output* untuk k_n sampel
$e[k]$ = *Error* dari k_n sampel = $Ref - ADC[k]$
$ADC[k]$ = Hasil konversi digital k_n sampel pada tegangan keluaran
Ref = Nilai digital sesuai dengan tegangan keluaran

DOI: https://doi.org/10.54082/jiki.4
P-ISSN: 2807-6664
E-ISSN: 2807-6591

$\sum_{i=0}^{k} e[i]$ = Akumulasi nilai *error*
$\{e[k] - e[k-1]\}$ = selisih *error* dari sampel k_n dengan k_{n-1}.

2.5. Fuzzy Logic Controller

Metode *fuzzy logic* pertama kali diusulkan oleh Lotfy Zadeh [12], dengan pendekatan menggunakan deskripsi linguistik dan sistem analisis berbasis teori dengan aturan yang samar (*fuzzy*). Jadi *fuzzy logic controller* (FLC) dapat diartikan sebagai sistem kendali yang berdasarkan logika *fuzzy*. Metode FLC ini telah banyak digunakan pada banyak alat, misalnya: alat pengukur kualitas air, sistem operasi kereta otomatis, kendali *elevator*, kendali reaktor nuklir, dan masih banyak lagi [13].



Gambar 2. Diagram Blok FLC

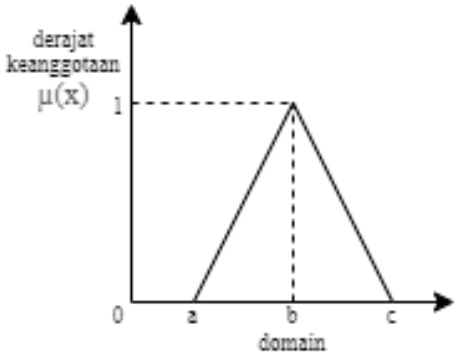
Dari blok diagram pada Gambar 2 dapat disimpulkan bahwa metode *fuzzy logic controller* memiliki tiga tahapan algoritma, yaitu:

a. Fuzzifikasi

Pada tahap fuzzifikasi, nilai masukan *crisp* atau pasti akan diubah menjadi nilai *fuzzy* yang berupa nilai linguistik. Nilai *fuzzy* dipengaruhi oleh fungsi keanggotaan [14]. Fungsi keanggotaan (*membership function)* adalah representasi pemetaan titik-titik *input* data ke dalam nilai/derajat keanggotaannya (memiliki nilai antara 0 sampai 1) dalam bentuk kurva. Untuk mendapatkan nilai keanggotaan dapat dilakukan dengan melakukan pendekatan fungsi. Ada beberapa fungsi yang dapat digunakan, contohnya sebagai berikut:

- Representasi Kurva Segitiga

Gambar 3 menunjukkan representasi *membership function* dengan menggunakan bentuk kurva segitiga. Nilai fungsi keanggotaan dapat dicari dengan persamaan (4).



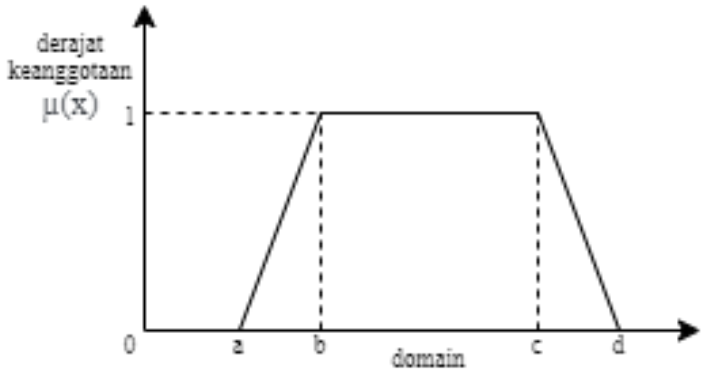
Gambar 3. Kurva Segitiga

DOI: https://doi.org/10.54082/jiki.4
P-ISSN: 2807-6664
E-ISSN: 2807-6591

$$\mu(x) = \begin{cases} 0; x \leq a \; atau \; x \geq c \\ \dfrac{x-a}{b-a}; a \leq x \leq b \\ \dfrac{b-x}{c-b}; b \leq x \leq c \end{cases} \quad (4)$$

$\mu(x)$ = nilai/derajat keanggotaan.
x = *range* domain nilai.
a, b dan c = nilai pada domain.

- Representasi Kurva Trapesium.



Gambar 4. Kurva Trapesium

Gambar 4 menunjukkan representasi *membership function* dengan menggunakan bentuk kurva trapesium. Nilai fungsi keanggotaan dapat dicari dengan persamaan (5).

$$\mu(x) = \begin{cases} 0; x \leq a \; atau \; x \geq d \\ \dfrac{x-a}{b-a}; a \leq x \leq b \\ 1; b \leq x \leq c \\ \dfrac{d-x}{d-c}; x \geq d \end{cases} \quad (5)$$

$\mu(x)$ = nilai/derajat keanggotaan.
x = *range* domain nilai.
a, b, c dan d = nilai pada domain.

b. Mekanisme *Inference*

Dalam proses *inference*, semua aturan yang ada dalam *rule-base* diperhitungkan, termasuk variabel masukan dan keluaran fungsi keanggotaan. Salah satu aturannya berupa logika "*if – then*" dapat dituliskan seperti *if* X_1 *is* A_1 *and …* *and* X_n *is* A_n , *then* Y *is* B. Dengan X_1, X_n dan Y adalah variabel linguistik yang merepresentasikan variabel keadaan proses sebanyak n dan satu variabel kontrol, serta A_1, A_n dan B sebagai variabel nilai [13], [14]. Aturan-aturan tersebut dibutuhkan untuk pengambilan keputusan akhir, yang nantinya menghasilkan keluaran *fuzzy*.

DOI: https://doi.org/10.54082/jiki.4
P-ISSN: 2807-6664
E-ISSN: 2807-6591

c. Defuzzifikasi

Pada tahap defuzzifikasi, nilai keluaran *fuzzy* tadi dikonversi kembali menjadi nilai keluaran *crisp* atau numerik yang paling sesuai dari nilai-nilai yang telah disimpulkan oleh keluaran linguistik. Pada metode Sugeno terdapat beberapa metode defuzzifikasi, antara lain:

- Metode *Weighted Average*

Pada metode *weighted average*, *output crisp* diperoleh dengan mengambil titik pusat daerah *fuzzy* (z^*). Secara umum dapat dirumuskan seperti pada persamaan (6).

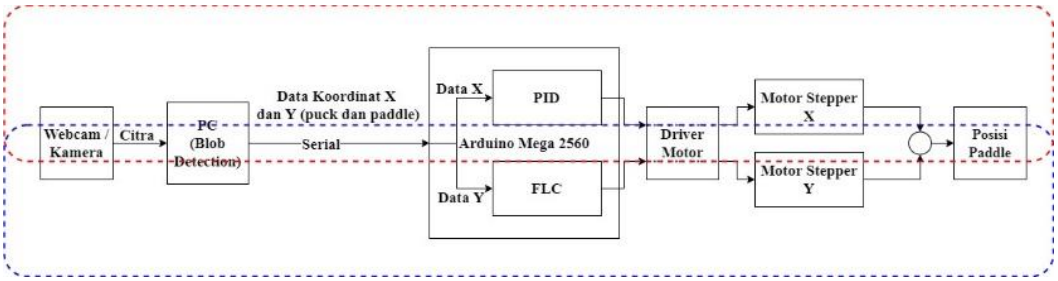
$$z^* = \frac{\sum_{i=1}^{n} z_i . w_i}{\sum_{i=1}^{n} w_i}; i = 1,2,3, \dots, n \quad (6)$$

n = Jumlah *rules*.
z = Nilai *output* dari *rules*, bisa berupa nilai konstan atau fungsi linear.
w = Bobot aturan yang berasal dari aturan sebelumnya.

- Metode *Weighted Sum*

Pada metode *weighted sum*, *output crisp* diperoleh dengan menjumlahkan seluruh nilai *output* dari *rules* yang telah dikalikan bobot *membership function output*-nya.

2.6. Desain Sistem Robot *Air Hockey*



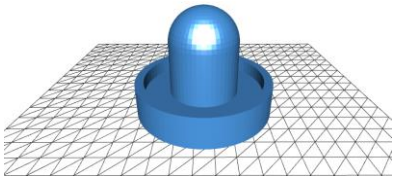
Gambar 5. Sistem Alat Keseluruhan

Gambar 5 menunjukkan alur proses kerja sistem secara menyeluruh. Sistem mengambil citra menggunakan *webcam* yang terhubung dengan *Personal Computer* (PC) lalu pengolahan citra menggunakan modul *blob detection* akan dilakukan. Hasil olahan citra akan berupa paket berisi data koordinat posisi *puck* dan *paddle* dalam sumbu X dan Y. Kemudian, data tersebut dikirim melalui komunikasi serial ke mikrokontroler (Arduino Mega 2560). Pada mikrokontroler, data koordinat pada sumbu X akan diproses kendali PID dan sumbu Y akan diolah menggunakan metode kendali FLC yang menghasilkan *output* posisi tujuan (*step*) untuk motor *stepper* Y. *Paddle* akan bergerak maju atau mundur seiring bergeraknya motor *stepper* Y yang bertujuan untuk memukul *puck* kembali ke sisi lawan.

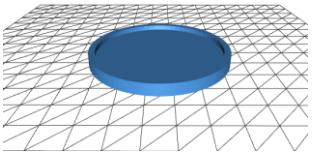
2.7. Desain Perangkat Keras

Sistem yang dibuat meliputi rancang bangun perangkat keras yang terdiri dari papan permainan *air hockey*, *puck*, dua buah *paddle* masing-masing dipakai untuk robot dan pemain, *timing belt* dan *pulley* yang akan terhubung ke motor stepper dan *paddle* robot, kipas yang ditanam di bawah papan guna mengurangi friksi dari *puck* dan *paddle*, dan motor stepper yang berlaku sebagai penggerak dari robot.

DOI: https://doi.org/10.54082/jiki.4
P-ISSN: 2807-6664
E-ISSN: 2807-6591

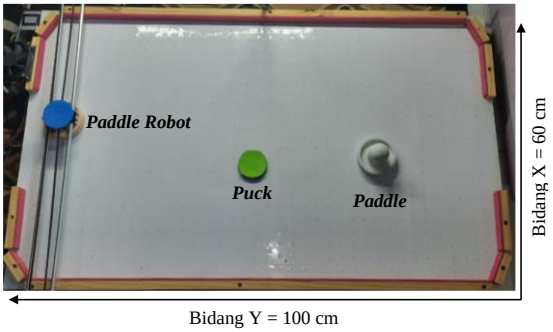


Gambar 6. Desain *Paddle*



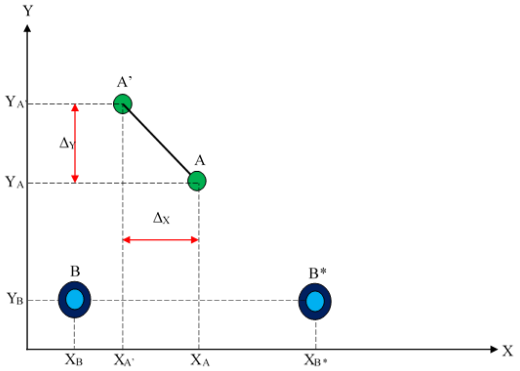
Gambar 7. Desain *Puck*

Gambar 6 menunjukkan desain dari *paddle* pada sistem yang dirancang dengan ukuran diameter 7 cm, Gambar 7 menunjukkan gambar dari desain *puck* yang digunakan pada sistem yang dirancang dengan ukuran diameter 6 cm. Gambar 8 menunjukkan tampak atas dari perangkat keras robot *air hockey*.



Gambar 8**.** Tampak Atas Perangkat Keras Robot *Air Hockey*

2.8. Desain Perangkat Lunak



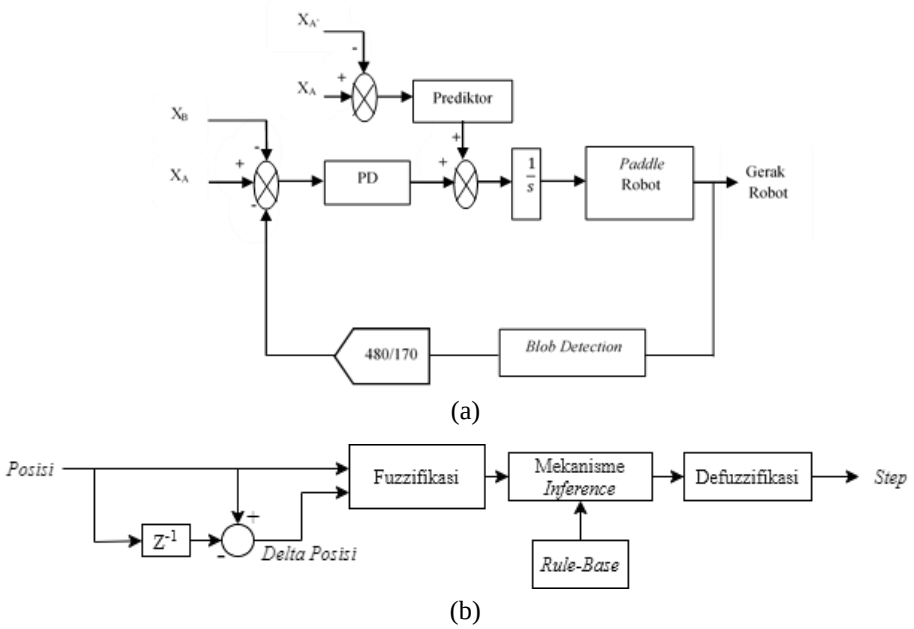
Gambar 9**.** Ilustrasi Data Gambar dari Kamera

A = Posisi *puck* actual.
Δ_y = Kecepatan *puck* pada komponen y.
Δ_x = Kecepatan *puck* pada komponen x.
B* = Titik tujuan gerak *paddle* robot.
B = Posisi robot aktual.

Pada Ilustrasi data gambar di Gambar 9, terdapat *puck* yang bergerak dari A' menuju A. A' merupakan data koordinat dari *frame* sebelumnya, dan A merupakan data dari *frame* aktual. B* merupakan estimasi titik potong antara *puck* dengan *paddle* robot sehingga titik B* akan dijadikan *set point* bagi sistem robot *air hockey*. Untuk mendapatkan titik ini, maka data yang digunakan adalah posisi aktual dari *puck* (A), posisi aktual dari robot (B) dan kecepatan *puck* pada komponen X (Δ_x). Posisi

DOI: https://doi.org/10.54082/jiki.4
P-ISSN: 2807-6664
E-ISSN: 2807-6591

aktual dari *puck* (A) dan robot (B) akan digunakan kendali PD untuk menentukan *set point* dalam proses *tracking puck*, sedangkan Δ_X akan digunakan prediktor untuk memprediksi titik potong *puck* dengan cara mengalikannya dengan suatu konstanta (K). Nilai konstanta akan dicari dengan metode *trial and error* atau teknik coba-coba. Nilai K yang dipilih adalah nilai yang dapat membuat gerak robot menjadi paling responsif dan optimal untuk menghalau *puck*. Diagram blok sistem PD terdapat pada Gambar 10 (a).



Gambar 10. Diagram Blok Sistem: (a) PID (b) FLC

Dalam merancang FLC pada sistem *air hockey* robot ini menggunakan metode Sugeno dengan dua *input* yaitu Posisi dan Delta Posisi dan *output* berupa *Step*. Metode Sugeno dipilih karena lebih efisien dan sesuai dengan *output* dari sistem ini yang berupa sebuah konstanta.

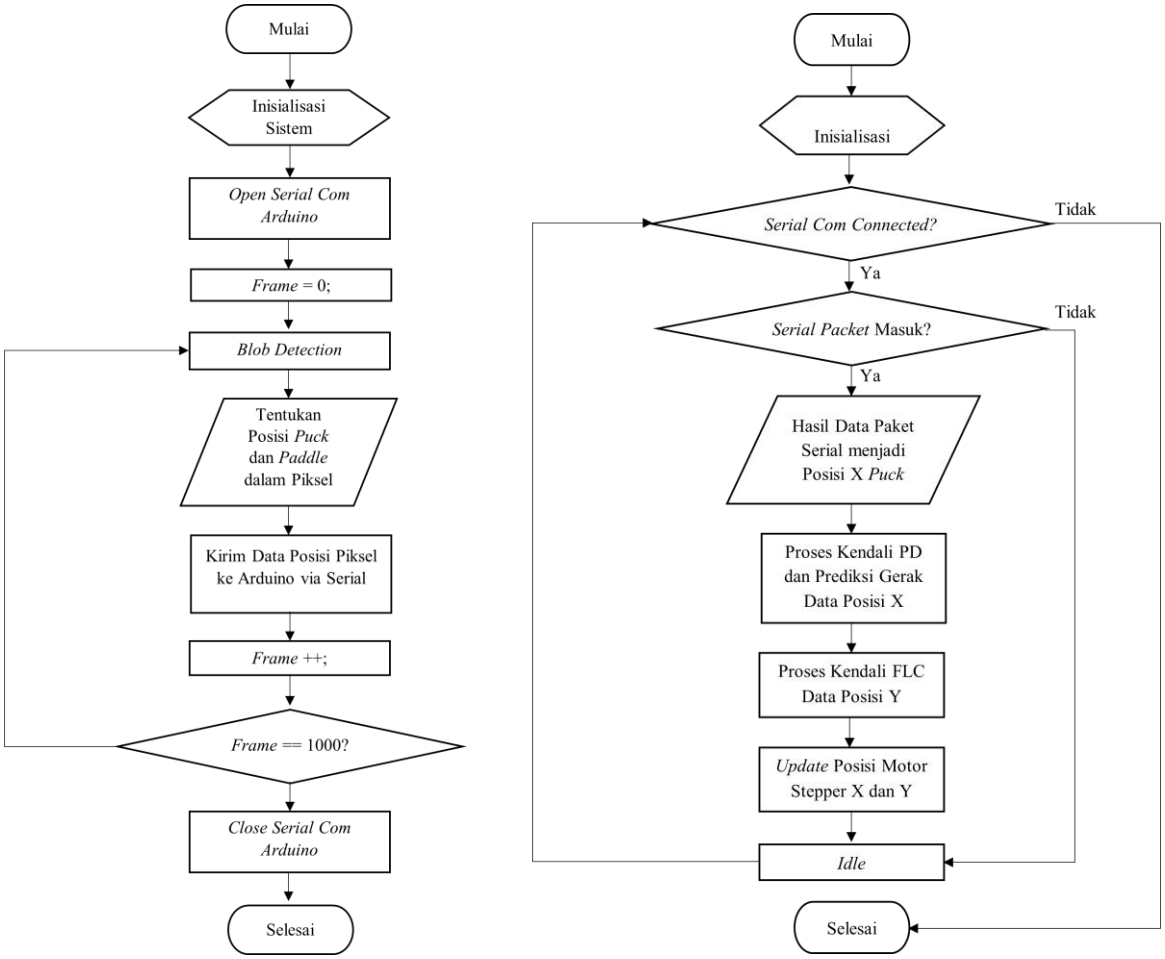
Gambar 10 (b) menunjukkan diagram blok dari sistem FLC yang dirancang. Sistem menggunakan dua buah *input* linguistik, yaitu 'Posisi' dan 'Delta Posisi'. *Input* posisi adalah letak *puck* pada meja *air hockey* dan delta posisi adalah hasil selisih *input* posisi aktual dikurang posisi sebelumnya. Dua *input* tersebut diolah melalui tahap fuzzifikasi, mekanisme *inference*, dan defuzzifikasi, lalu menghasilkan *output* berupa nilai *step* yang digunakan untuk mengatur gerakan motor *stepper*. Sistem FLC pada *air hockey robot* mempunyai *rule-base* seperti pada Tabel 3.

Tabel 3. *Rule-Base* FLC

		Delta Posisi		
		Negatif	*Zero*	Positif
Posisi	Lawan	*Low*	*Low*	*Low*
	Tengah	*Low*	*Medium*	*High*
	Robot	*High*	*High*	*High*

Pada sistem *air hockey robot* terdapat dua program, yaitu program untuk sistem pengolahan citra dengan menggunakan MATLAB dan program untuk sistem kendali posisi yang berada pada Arduino. Gambar 11 menunjukkan *flowchart* untuk masing-masing sistem.

DOI: https://doi.org/10.54082/jiki.4
P-ISSN: 2807-6664
E-ISSN: 2807-6591



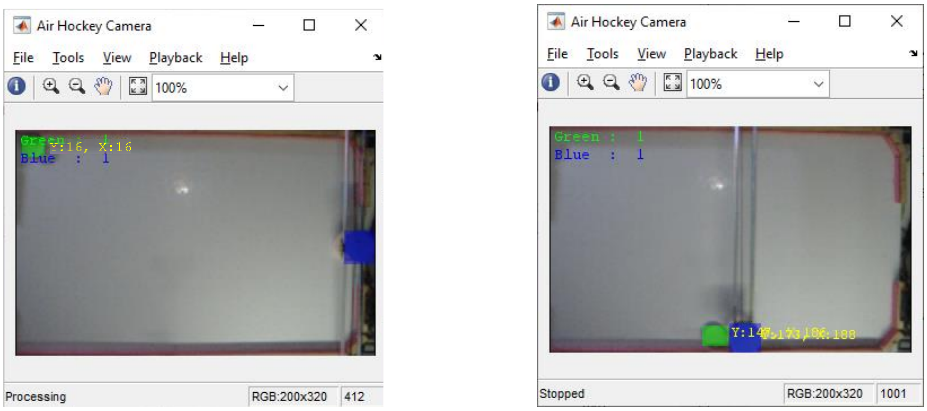
Gambar 11. *Flowchart* Sistem: (a) Pengolahan Citra (b) Kendali Posisi *Paddle* Robot

3. HASIL DAN PEMBAHASAN

3.1. Pengujian Pengolahan Citra

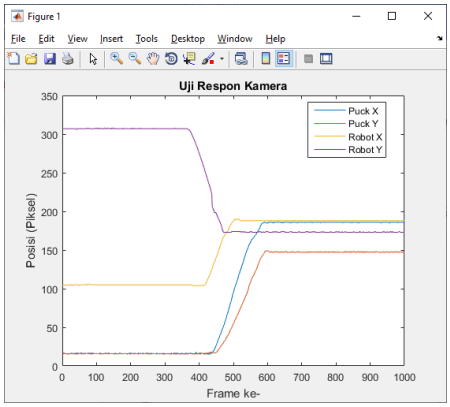
Pada bagian ini, sensor kamera akan diuji ketepatannya serta akan dihitung berapa lama waktu komputasi yang diperlukan oleh sistem untuk melakukan pengolahan citra hingga dihasilkan keluaran berupa data posisi piksel yang siap digunakan. Prosedur pengujian akan dilakukan dengan mengubah posisi *puck* dengan warna hijau dan robot dengan warna biru ke dua titik berbeda. *Puck* akan ditempatkan pada dua posisi dengan nilai koordinat (X, Y) posisi awal adalah (16, 16) dan posisi akhirnya (186,147), robot akan memiliki posisi awal (105, 137) dan posisi akhir (188,173). *Puck* dan robot akan dipantau menggunakan kamera yang ditampilkan oleh MATLAB. Nilai ini akan dicocokkan dengan data keluaran piksel berupa grafik. Dari grafik pula akan dihitung seberapa cepat waktu komputasi yang dilakukan untuk melakukan pengolahan citra pada setiap *frame*. Hasil pengujian kamera dapat dilihat melalui Gambar 12 dan Gambar 13.

DOI: https://doi.org/10.54082/jiki.4
P-ISSN: 2807-6664
E-ISSN: 2807-6591



(a) (b)

Gambar 12. Koordinat Posisi: (a) Awal, (b) Akhir

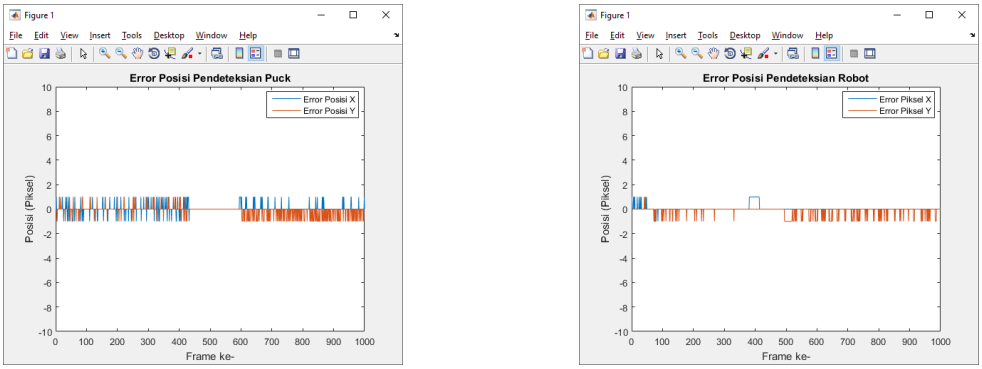


Gambar 13. Grafik Respons Pengujian Kamera

Pada pengujian sensor kamera yang ditunjukkan Gambar 13, terdapat 1000 sampel *frame* yang didapat dengan durasi waktu 18 detik, 55 FPS. Dari hasil tersebut dapat diperoleh bahwa pemrosesan citra yang dilakukan untuk menuntaskan satu *frame* adalah 0.018 detik. Waktu pemrosesan citra ini dapat ditingkatkan performanya dengan cara mematikan *real time video player* pada MATLAB, sehingga Gambar 12 tidak akan ditampilkan ketika program berjalan. Apabila hal tersebut dilakukan, *framerate* dapat mencapai 80 FPS, yang artinya untuk menuntaskan satu *frame* dalam pengolahan citra dibutuhkan waktu selama 0,0125 detik. Waktu tersebut sangat memadai untuk mendukung gerak robot yang dituntut untuk bergerak responsif. Dengan demikian, opsi mematikan *real time video player* pada MATLAB akan ditempuh.

Pada Gambar 13 dapat dilihat hasil pembacaan koordinat dalam piksel yang secara garis besar cukup sesuai. Nilai koordinat *puck* dan robot akan menuju ke dua titik utama yang merupakan posisi tetap yang telah ditentukan dan akan bertransisi seiring dengan berpindahnya posisi *puck* dan robot. Ketika *puck* berada diam pada posisi tetapnya, terjadi sedikit *jitter* dengan nilai *error* maksimum adalah ±1 piksel. Terjadinya *error* piksel ini kemungkinan besar dihasilkan oleh getaran yang terjadi pada alat dan mempengaruhi kondisi kamera yang berada di atasnya sehingga kamera tidak dalam keadaan yang *steady*. Dengan dua posisi berbeda, hasil data keluaran piksel yang didapat pada Gambar 13 menunjukkan total nilai akurasi 93% dengan nilai *error* maksimum per sampel ± 1 piksel. Grafik *error* posisi dalam piksel dapat dilihat pada Gambar 14. *Error jitter* ditunjukkan apabila grafik menunjukkan nilai yang tidak nol.

DOI: https://doi.org/10.54082/jiki.4
P-ISSN: 2807-6664
E-ISSN: 2807-6591



(a) (b)

Gambar 14. Grafik *Error* Piksel Pendeteksian: (a) *Puck*, (b) Robot

3.2. Pengujian Statis

a. Bidang X

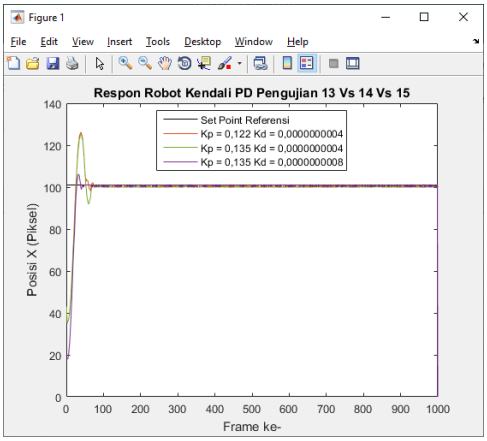
Prosedur yang dilakukan adalah dengan cara menentukan *set point* tetap untuk robot dan membiarkan robot untuk memberikan responsnya terhadap referensi tersebut. Metode yang ditempuh untuk mendapatkan parameter PID adalah *trial and error*. Parameter K_p dan K_d yang akan diujikan dapat dilihat pada Tabel 4.

Tabel 4. Pengujian Kendali PD *Fine Tuning*

Pengujian ke-	K_p	K_d	Respons Robot
1	0,244	4	Osilasi
2	0,244	0,4	Osilasi
3	0,244	0,04	Steady
4	0,244	0,004	Steady
5	0,244	0,0004	Steady
6	0,244	0,00004	Steady
7	0,244	0,000004	Steady
8	0,244	0,0000004	Steady
9	0,244	0,00000004	Steady
10	0,244	0,000000004	Steady
11	0,244	0,0000000004	Steady
12	0,0244	0,0000000004	SP tidak tercapai
13	0,122	0,0000000004	Steady
14	0,135	0,0000000004	Steady
15	0,135	0,0000000008	Steady

Berdasarkan Tabel 4 di atas, respons robot yang menghasilkan gerakan cukup baik terjadi pada pengujian 13, 14 dan 15 yang ditampilkan pada Gambar 15. Pada pengujian 12 diperoleh titik *threshold* terkecil dari parameter yang diujikan yang ditandai dengan tidak tercapainya *set point*. Berdasarkan data tersebut, nilai parameter K_p dan K_d dinaikkan hingga ditemukan respons yang paling optimal bagi pergerakan *paddle* robot.

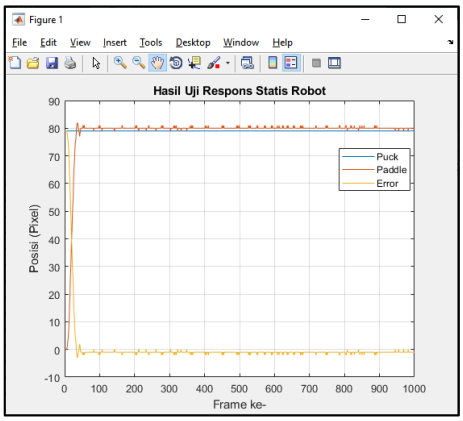
DOI: https://doi.org/10.54082/jiki.4
P-ISSN: 2807-6664
E-ISSN: 2807-6591



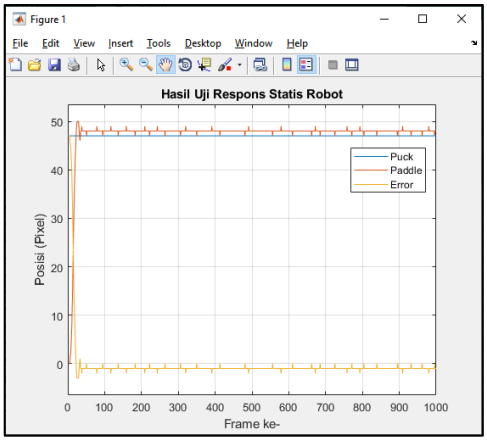
Gambar 15. Grafik Respons Robot Pengujian 13, 14 dan 15

Berdasarkan Gambar 15, kendali PD dengan $K_p = 0{,}135$ dan $K_d = 0{,}0000000008$ akan dipilih karena memiliki *overshoot* yang lebih kecil 19% dibandingkan dua yang lain. Dengan kendali PD $K_p = 0{,}135$ dan $K_d = 0{,}0000000008$ dapat memberikan performa *rise time* 0,35 detik, *overshoot* 2,94%, *settling time* 0,5 detik dan *error steady state* 0,58%. Kendali dengan nilai $K_p = 0{,}135$ dan $K_d = 0{,}0000000008$ ini akan digunakan untuk kepentingan pengujian selanjutnya.

b. Bidang Y



Gambar 16. Grafik Hasil Uji Statis ke-1



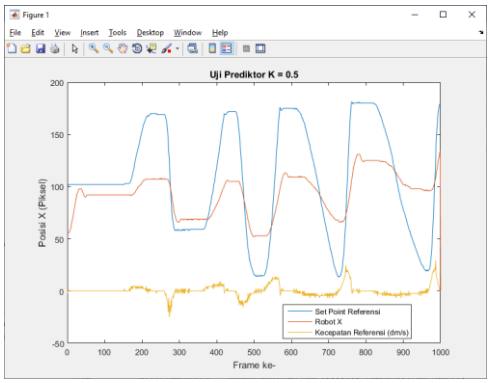
Gambar 17. Grafik Hasil Uji Statis ke-2

DOI: https://doi.org/10.54082/jiki.4
P-ISSN: 2807-6664
E-ISSN: 2807-6591

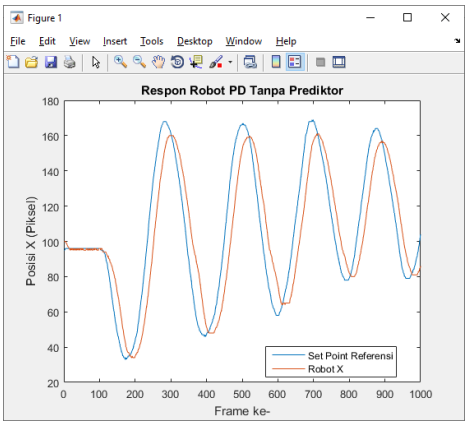
Gambar 16 menunjukkan grafik respons robot terhadap *puck* menggunakan FLC yang diletakkan pada posisi di titik 79 *pixel*, terlihat bahwa *paddle* bergerak maju menuju posisi *puck* dengan memiliki nilai *overshoot* sebesar 3,80% dan *error steady state* sebesar 1,27%, dengan *rise time* sebesar 0,425 detik. Pada Gambar 17 menunjukkan grafik respons robot terhadap *puck* yang diletakkan pada posisi di titik 47 *pixel*, terlihat bahwa *paddle* bergerak maju menuju posisi *puck* dengan memiliki nilai *overshoot* sebesar 6,38% dan *error steady state* sebesar 2,13%, dengan *rise time* sebesar 0,287 detik.

Dari hasil pengujian dapat dilihat bahwa robot mampu menggerakkan *paddle* ke posisi *puck* dan mempertahankannya dengan nilai maksimum *error steady state* sebesar 2,13%. Robot juga dapat merespons dengan cepat dalam waktu paling lambat di 0,425 detik.

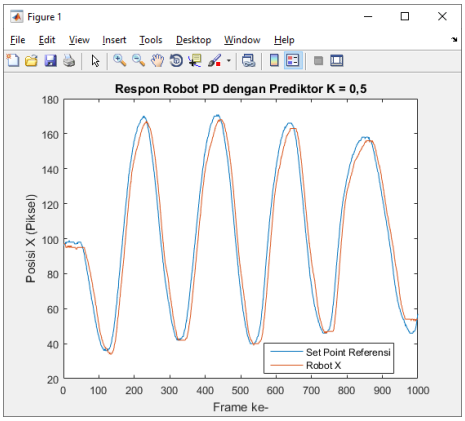
3.3. Pengujian Prediktor



Gambar 18. Grafik Respons Uji Prediktor K= 0,5



Gambar 19. Respons Robot Tanpa Prediktor



Gambar 20. Respons Robot dengan Prediktor

Prediktor digunakan agar gerakan *tracking* yang dilakukan robot oleh kendali PD $K_p = 0{,}135$ dan $K_d = 0{,}0000000008$ pada bagian sebelumnya dapat dikompromikan sehingga gerak robot yang dihasilkan tetap responsif terhadap serangan yang datang. Pada kendali PD, robot akan berusaha melakukan *tracking* pada referensi yang diberikan sehingga kemungkinan buruk pada gerak robot, seperti tertinggalnya robot terhadap cepatnya serangan mungkin saja terjadi, tetapi dengan disematkannya algoritma prediktor, robot diizinkan untuk bergerak selangkah lebih maju. Bila referensi yang digunakan oleh kendali PD pada dasarnya adalah posisi, maka referensi yang digunakan oleh prediktor adalah kecepatan dari referensi. Prediktor dengan K = 0,5 akan digunakan pada sistem robot

DOI: https://doi.org/10.54082/jiki.4
P-ISSN: 2807-6664
E-ISSN: 2807-6591

air hockey karena memiliki kemampuan prediksi yang optimal. Ketika prediktor dengan K = 0,2 tidak terlalu peka terhadap perubahan kecepatan referensi dan prediktor dengan K = 0,9 terlalu agresif terhadap perubahan kecepatan referensi, prediktor dengan K = 0,5 menjadi *sweet spot* untuk pergerakan robot *air hockey*.

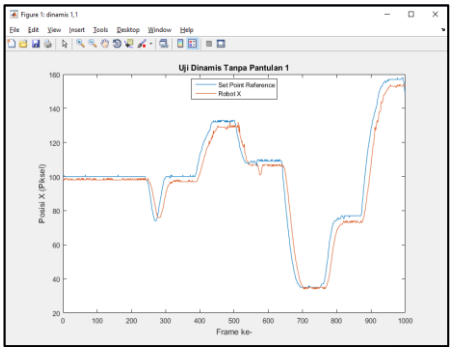
Gambar 18 menunjukkan hasil pengujian algoritma prediktor tanpa adanya kendali PD, berdasarkan grafik yang ditampilkan, *paddle* robot akan bergerak sesuai dengan keadaan kecepatan yang terjadi pada *puck*, bukan dari posisi aktual *puck.*

Gambar 19 dan Gambar 20 menunjukkan perbedaan dari sistem kendali gerak robot antara kendali PD saja dengan kendali PD yang disematkan algoritma prediktor. Kendali PD yang disematkan algoritma prediktor memiliki respons yang lebih baik, ditandai dengan kemampuannya untuk mencapai *set point* yang lebih cepat dan memiliki kemampuan untuk mengurangi *delay* yang terjadi untuk mencapai *set point*.

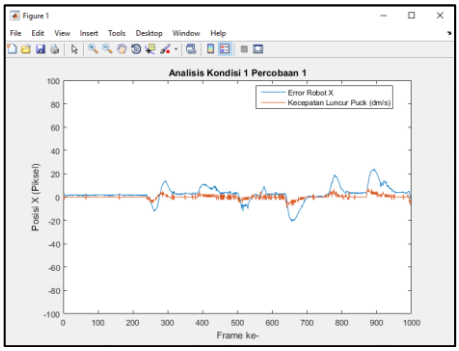
3.4. Pengujian Robot *Air Hockey*

Pada bagian ini, keseluruhan sistem *air hockey* diuji secara dinamis. Tujuan dari pengujian ini adalah menempatkan robot dalam situasi layaknya permainan *air hockey* yang sebenarnya. Dengan demikian, akan terjadi perubahan *set point* yang terus menerus secara dinamis seiring bergeraknya *puck.* Prosedur pengujiannya dengan memberikan serangan kepada robot secara terus menerus.

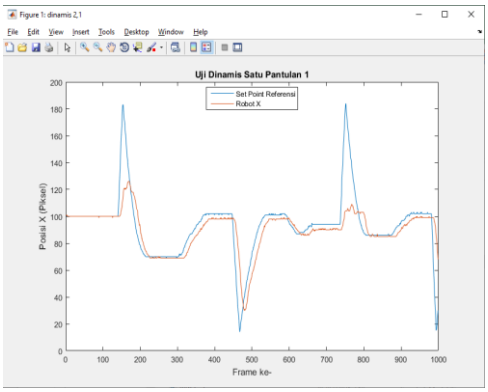
a. Bidang X



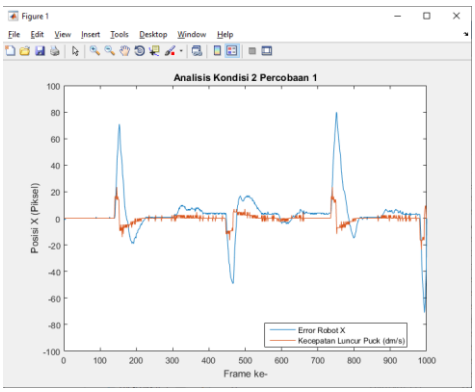
Gambar 21. Grafik Respons Uji Dinamis Kondisi 1



Gambar 22. Grafik Analisis Kondisi 1



Gambar 23. Grafik Respons Uji Dinamis Kondisi 2



Gambar 24. Grafik Analisis Kondisi 2

Pada bagian ini, keseluruhan sistem *air hockey* diuji secara dinamis. Tujuan dari pengujian ini adalah menempatkan robot dalam situasi layaknya permainan *air hockey* yang sebenarnya. Dengan demikian, akan terjadi perubahan *set point* yang terus menerus secara dinamis seiring bergeraknya *puck.*

DOI: https://doi.org/10.54082/jiki.4
P-ISSN: 2807-6664
E-ISSN: 2807-6591

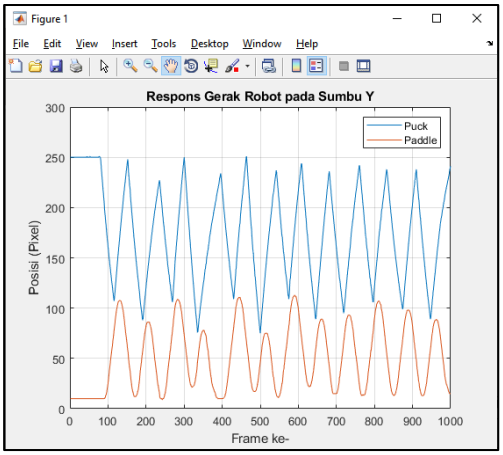
Prosedur pengujiannya dengan memberikan serangan kepada robot secara terus menerus dengan memenuhi dua kondisi, kondisi pertama adalah melakukan serangan langsung tanpa pantulan dan kondisi kedua adalah melakukan serangan dengan sekali pantulan. Kecepatan serangan untuk meluncurkan *puck* dilakukan secara variatif dengan tujuan menguji sejauh mana kemampuan robot mampu menghalau serangan yang datang. Pada pengujian ini digunakan kendali dengan respons terbaik, yaitu dengan kendali PD *fine tuning* $K_p = 0{,}135$, $K_d = 0{,}0000000008$ dan prediktor K = 0,5.

Pada Gambar 21 dan Gambar 22 menampilkan hasil percobaan pada pengujian kondisi 1, sedangkan Gambar 23 dan Gambar 24 menampilkan hasil percobaan pada pengujian kondisi 2, grafik biru merepresentasikan gerak dari *puck* sekaligus menjadi referensi gerak bagi *paddle* robot, sedangkan grafik coklat adalah respons gerak *paddle* robot pada sumbu X yang dikendalikan dengan kendali PD dan algoritma prediktor. Pengujian dinamis robot *air hockey* dapat menghasilkan tingkat respons terbaik pada skenario kondisi serangan tanpa pantulan dengan persentase *error* rata-rata robot X adalah 3,44% dengan kecepatan luncur maksimum *puck* pada vektor X rata-rata adalah 1,01 m/s. Skenario pengujian satu pantulan menghasilkan persentase *error* rata-rata robot X adalah 6,11% dengan kecepatan luncur maksimum *puck* pada vektor X rata-rata adalah 2,44 m/s.

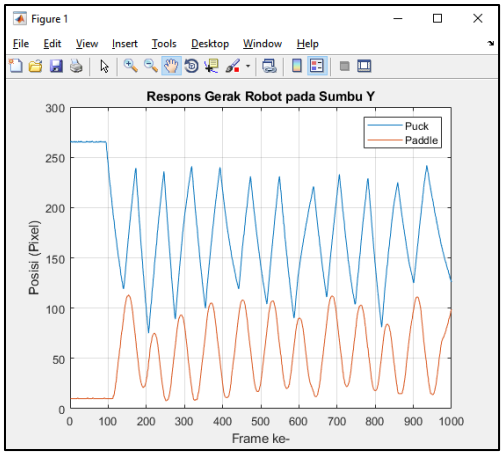
b. Bidang Y

Pada tahap ini pengujian robot dilakukan dengan cara bermain *air hockey* seperti pada umumnya, yaitu dengan memukul *puck* ke arah robot, kemudian dilihat responsnya apakah robot berhasil mengembalikan *puck* ke sisi lawan, jika titik balik *puck* bernilai > 173 pixel maka robot dapat dikatakan berhasil mengembalikan *puck* ke sisi lawan.

Gambar 23 dan Gambar 24 menunjukkan grafik pergerakan *puck* dan *paddle*, grafik yang naik turun menunjukkan bahwa *puck* dan *paddle* bergerak maju dan mundur. Terlihat juga bahwa saat kurva *puck* menurun maka kurva *paddle* mengalami kenaikan, hal tersebut menunjukkan bahwa robot merespons kedatangan *puck* dengan cara memajukan *paddle* supaya *puck* terpukul.



Gambar 25. Grafik Hasil Uji Dinamis ke-1



Gambar 26. Grafik Hasil Uji Dinamis ke-2

Tabel 5 dan 6 berisi data hasil uji dinamis pada *air hockey robot*, terdapat titik ketika *user* memukul *puck* (titik pergi), titik ketika *puck* dipukul oleh *paddle* (titik pukul), titik ketika *puck* dipukul kembali oleh *user* (titik pulang), serta kecepatan *puck* saat pergi dan pulang. Pada pengujian ke-1, dalam 1000 *frame* terdapat 12 pukulan sedangkan pada pengujian ke-2 terdapat 11 pukulan dengan titik pergi, titik pukul, dan titik pulang yang bervariasi. Terlihat juga dari data yang ditunjukkan oleh Tabel 2 dan 3,

DOI: https://doi.org/10.54082/jiki.4
P-ISSN: 2807-6664
E-ISSN: 2807-6591

robot dapat merespons datangnya *puck* dengan baik, hal itu ditunjukkan dengan berhasilnya robot mengembalikan *puck* ke sisi lawan.

Tabel 5. Data Hasil Uji Dinamis ke-1

Pukulan ke-	Titik Pergi *Puck* *(Pixel)*	Titik Pukul *Puck* *(Pixel)*	Titik Pulang *Puck* *(Pixel)*	Kecepatan Maks. Pergi *Puck* (m/s)	Kecepatan Maks. Pulang Puck (m/s)	Ket.
1	250	107	248	1,59	1,13	Berhasil
2	248	88	226	1,82	1,13	Berhasil
3	226	106	250	1,13	1,82	Berhasil
4	250	76	234	1,36	1,13	Berhasil
5	234	109	250	1,13	1,59	Berhasil
6	250	75	235	1,82	1,13	Berhasil
7	235	109	243	1,36	2,27	Berhasil
8	243	89	236	1,59	1,36	Berhasil
9	236	95	241	1,13	1,36	Berhasil
10	241	106	238	1,36	1,59	Berhasil
11	238	99	238	1,13	1,13	Berhasil
12	238	89	241	1,59	1,13	Berhasil

Tabel 6. Data Hasil Uji Dinamis ke-2

Pukulan ke-	Titik Pergi *Puck* *(Pixel)*	Titik Pukul *Puck* *(Pixel)*	Titik Pulang *Puck* *(Pixel)*	Kecepatan Maks. Pergi *Puck* (m/s)	Kecepatan Maks. Pulang *Puck* (m/s)	Ket.
1	265	119	239	1,59	1,13	Berhasil
2	239	79	236	2,04	1,59	Berhasil
3	236	89	241	1,59	1,13	Berhasil
4	241	100	240	1,59	1,36	Berhasil
5	240	119	231	0,91	1,36	Berhasil
6	231	104	231	0,91	1,13	Berhasil
7	231	90	221	1,59	1,13	Berhasil
8	221	112	233	1,13	1,13	Berhasil
9	233	104	229	1,36	1,36	Berhasil
10	229	81	225	1,59	1,59	Berhasil
11	225	125	240	0,68	1,13	Berhasil

Pada pengujian ke-1 robot terlihat mampu merespons datangnya *puck* dengan kecepatan maksimum sebesar 1,82 m/s dan robot berhasil mengembalikan *puck* dengan kecepatan maksimum 2,27 m/s. Sedangkan pada pengujian ke-2 robot mampu merespons datangnya *puck* dengan kecepatan maksimum sebesar 2,04 m/s, robot cenderung mengembalikan *puck* lebih lambat dari kecepatan datangnya, namun pada pukulan ke-5, 6, dan 11 robot dapat mengembalikan *puck* lebih cepat dari kecepatan datangnya. Robot juga dapat mengembalikan *puck* sama dengan kecepatan datangnya dengan kecepatan maksimum sebesar 1,59 m/s.

Dari hasil pengujian 1 dan 2, grafik menunjukkan bahwa *puck* dan *paddle* bergerak maju dan mundur secara berlawanan. Hal tersebut menunjukkan bahwa robot merespons datangnya *puck* dengan memajukan *paddle* untuk memukul *puck*. Robot mampu merespons *puck* yang bergerak dengan kecepatan maksimum sebesar 2,04 m/s dan berhasil mengembalikan *puck* dengan kecepatan maksimum

DOI: https://doi.org/10.54082/jiki.4
P-ISSN: 2807-6664
E-ISSN: 2807-6591

sebesar 2,27 m/s. Kecepatan *puck* yang dikembalikan oleh robot terkadang lebih lambat, sama, bahkan lebih cepat dari kecepatan datangnya *puck.* Hal tersebut dapat disebabkan karena momentum saat terjadinya pukulan pada *puck* oleh *paddle*.

Hasil pengujian dinamis yang telah dilakukan baik pada bidang X dan bidang Y menunjukkan bahwa robot berhasil menangani seluruh serangan yang dilakukan terhadap robot, baik serangan secara langsung tanpa pantulan maupun serangan dengan sekali pantulan. Dengan beragam kecepatan serangan *puck* yang diberikan terhadap robot, robot berhasil mengembalikan *puck* lebih lambat, sama, bahkan lebih cepat dari kecepatan datangnya *puck*.

4. KESIMPULAN

Berdasarkan hasil pengujian dan analisis implementasi PID dan FLC berbasis *image processing* pada *air hockey* robot, maka dapat diambil kesimpulan bahwa robot berhasil mendeteksi posisi *puck* dan *paddle* dengan menggunakan metode *blob detection* pada *image processing* dengan tingkat akurasi sebesar 93%.

Pada pengujian statis, kendali PD yang bernilai parameter K_p = 0,135, K_d = 0,0000000008 mampu memberikan respons dengan *rise time* 0,35 detik, *overshoot* 2,94%, *settling time* 0,5 detik dan *error steady state* 0,58%. Sistem kendali FLC mampu menggerakkan *paddle* ke posisi *puck* dengan *error steady state* maksimum sebesar 2,13%. Waktu respons keseluruhan sistem *air hockey yang* dimulai dari penangkapan gambar oleh kamera hingga robot berada di fase *steady state* adalah 0,5125 detik.

Pada pengujian dinamis, penerapan kendali PD dan prediktor K = 0,5 menghasilkan respons terbaik pada skenario kondisi serangan tanpa pantulan dengan persentase *error* rata-rata robot dalam bidang X adalah 3,44%. Pada bidang Y, Robot dengan implementasi FLC mampu merespons dengan baik datangnya *puck*, hal ini ditunjukkan oleh bentuk grafik dan berhasilnya robot mengembalikan *puck* ke sisi lawan. Robot juga mampu merespons *puck* yang datang dengan kecepatan maksimum 2,04 m/s dan mengembalikannya dengan kecepatan maksimum sebesar 2,27 m/s.

DAFTAR PUSTAKA

[1] H. D. Siswaja, "Prinsip Kerja dan Klasifikasi Robot," *Media Inform.*, vol. 7, no. 3, pp. 147–157, 2008.

[2] D. Silver *et al.*, "A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play," vol. 1144, no. December, pp. 1140–1144, 2018, doi: 10.1126/science.aar6404.

[3] A. Namiki and T. Ozeki, "Vision-Based Optimal Control for an Air-Hockey Robot System," *2017 IEEE 7th Annu. Int. Conf. CYBER Technol. Autom. Control. Intell. Syst.*, pp. 1176–1181, 2017, doi: 978-1-5386-0490-8/17.

[4] C. Berntsson, L. Brown, S. Fitz, A. Hallberg, D. Sondell, and K. Svensson, "Autonomous Air Hockey," *Bachelor's Thesis*, p. 44, 2016, [Online]. Available: https://hdl.handle.net/20.500.12380/242068.

[5] M. Ansgar, H. Berggren, P. Borg, R. Damberg, M. Gudjonsson, and L. Mared, "Autonomous Air Hockey," *Bachelor's Thesis*, p. 38, 2016, [Online]. Available: https://hdl.handle.net/20.500.12380/240634.

[6] Efti, "Air hockey table size chart – All types of table," *PROEIST SPORT*, 2020. https://proiest.com/air-hockey-table-size.

[7] C. R. Nurhuda and K. Firdausy, "Metode Color Blob Detection Untuk Deteksi Kematangan Tomat Secara Otomatis," pp. 405–410, 2017.

[8] L. Wang, *PID Control System Design and Automatic Tuning using MATLAB/Simulink*. 2020, doi: 10.1002/9781119469414.

DOI: https://doi.org/10.54082/jiki.4
P-ISSN: 2807-6664
E-ISSN: 2807-6591

[9] M. Ali, "Pembelajaran Perancangan Sistem Kontrol Pid Dengan Software Matlab," *J. Edukasi Elektro*, vol. 1, no. 1, p. 2, 2004.
[10] C. R. Phillips and N. T. Nagle, *Digital control system analysis and design*, vol. SMC-15, no. 3. 2012, doi: 10.1109/TSMC.1985.6313385.
[11] A. Visioli, *Practical PID Control*. 2006, doi: 10.1007/1-84628-586-0.
[12] L. A. Zadeh, "Fuzzy Sets," *Inf. Control*, no. 8, pp. 338–353, 1985, doi: 10.1061/9780784413616.194.
[13] C. C. Lee, "Fuzzy Logic in Control Systems: Fuzzy Logic Controller, Part II," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 20, no. 2. pp. 419–435, 1990, doi: 10.1109/21.52552.
[14] . R., K. Exaudi, and A. P. P. Prasetyo, "Navigasi Berbasis Behavior dan Fuzzy Logic pada Simulasi Robot Bergerak Otonom," *J. Nas. Tek. Elektro*, vol. 5, no. 1, 2016, doi: 10.20449/jnte.v5i1.212.

DOI: https://doi.org/10.54082/jiki.4
P-ISSN: 2807-6664
E-ISSN: 2807-6591

Halaman Ini Dikosongkan