

# Penerapan Image Processing Dalam Sistem Monitoring Ketinggian Air

**Ikhsan Ferdiansyah<sup>a1</sup>, Anak Agung Ketut Agung Cahyawan Wiranatha<sup>a2</sup>,  
I Made Sunia Raharja<sup>a3</sup>**

<sup>a</sup>Program Studi Teknologi Informasi, Fakultas Teknik, Universitas Udayana  
Bukit Jimbaran, Bali, Indonesia Telp. (0361) 701806

e-mail: [1ikhsan\\_ferdiansyah@student.unud.ac.id](mailto:1ikhsan_ferdiansyah@student.unud.ac.id), [2agung.cahyawan@unud.ac.id](mailto:2agung.cahyawan@unud.ac.id),  
[3sunia.raharja@unud.ac.id](mailto:3sunia.raharja@unud.ac.id)

## **Abstrak**

*Banjir adalah bencana alam yang terjadi ketika volume air melebihi kapasitas sungai atau sistem aliran air. Salah satu cara untuk menentukan apakah volume air masih dapat diakomodasi adalah dengan mengukur ketinggian air. Penelitian ini mengaplikasikan metode Image Processing untuk mengukur ketinggian air yang bekerja dengan mengukur jarak antara pelampung dengan kamera. Deteksi pelampung dilakukan menggunakan Algoritma Hough Circle Transform, dan pengukuran jarak menggunakan Persamaan Segitiga. Hasil pengujian menunjukkan bahwa Algoritma Hough Circle Transform dapat mendeteksi pelampung dengan diameter 38 mm pada jarak minimum 10 cm dan jarak maksimum 60 cm. Pengukuran ketinggian air menggunakan Persamaan Segitiga menunjukkan korelasi yang kuat (0,999) antara jarak yang diukur dan jarak sebenarnya.*

**Kata kunci:** IoT, Hough Circle Transform, Persamaan Segitiga, Raspberry Pi

## **Abstract**

*Flood is a natural disaster that occurs when the volume of water exceeds the capacity of a river or water flow system. One way to determine if the water volume can still be accommodated is by measuring the water level. This research applies Image Processing methods to measure water level by measuring the distance between a float and a camera. The float detection is performed using the Hough Circle Transform Algorithm, and the distance measurement uses the Triangle Equation. The test results show that the Hough Circle Transform Algorithm can detect floats with a diameter of 38 mm at a minimum distance of 10 cm and a maximum distance of 60 cm. The water level measurement using the Triangle Equation shows a strong correlation (0.999) between the measured distance and the actual distance.*

**Keywords :** IoT, Hough Circle Transform, Triangle Similarity, Raspberry Pi

## **1. Introduction**

Bencana alam menurut KBBI adalah bencana yang disebabkan oleh alam. Bencana itu sendiri merupakan sesuatu yang menyebabkan kerusakan, kerugian atau penderitaan, maka bencana alam dapat diartikan sebagai kerusakan, kerugian atau penderitaan yang disebabkan oleh alam. Serangkaian bencana yang disebabkan oleh alam ini antara lain berupa gempa bumi, gunung meletus, tsunami, angin topan, tanah longsor dan banjir [1].

Indonesia sebagai negara beriklim tropis yang berarti hanya memiliki dua musim yaitu musim penghujan dan musim kemarau. Keadaan ini memberikan keuntungan untuk beberapa sektor di Indonesia, salah satunya adalah sektor pertanian. Musim penghujan membuat tanah menjadi subur sehingga baik untuk perkebunan dan pertanian di Indonesia. Dibalik kelebihan yang dimiliki, musim penghujan ini juga memiliki kerugian yang dapat dirasakan, salah satunya dapat menimbulkan bencana alam berupa banjir.

Hujan dengan intensitas tinggi dapat meningkatkan volume air, sehingga meningkatkan potensi banjir. Volume air yang melebihi daya tampung ini dapat menyebabkan terjadinya banjir. Bencana banjir tentunya membawa kerugian bagi masyarakat sekitar bahkan dapat menelan korban jiwa [2]. Salah satu solusi untuk mengetahui potensi banjir pada suatu daerah adalah

memantau volume atau ketinggian air pada sungai. Pemantauan yang akurat dan *real-time* dapat memperkirakan potensi banjir sehingga pemberitahuan dini juga dapat dilakukan untuk mencegah jatuhnya korban akibat bencana banjir ini.

Banjir terjadi ketika volume air pada sebuah saluran air lebih tinggi dari kapasitas yang ditampungnya sehingga terjadi luapan. Luapan ini dapat dilihat dari air yang naik melewati saluran air. Salah satu cara untuk memprediksi potensi banjir adalah mengukur ketinggian permukaan air. Sudah ada beberapa penelitian terkait pengukuran tinggi permukaan air contohnya adalah penelitian yang dilakukan oleh [3] yang melakukan pengukuran ketinggian air menggunakan sensor *ultrasonic*. Berdasarkan 100 data pengujian, terdapat lima kesalahan dengan nilai 1 cm dan satu kesalahan dengan nilai 19 cm.

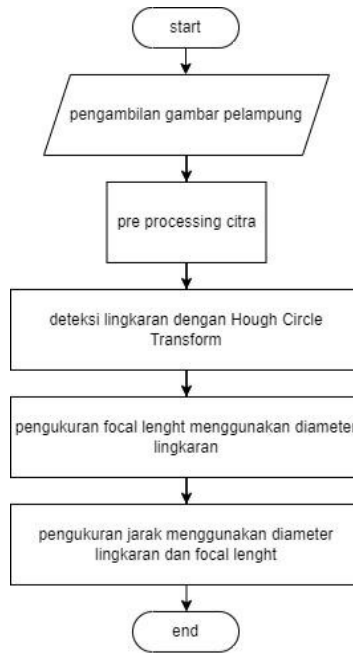
Pengukuran ketinggian air dapat dilakukan menggunakan alat ukur sederhana seperti meteran. Pengukuran dengan cara ini tentunya kurang efektif untuk memantau ketinggian air, karena ketinggian permukaan air yang dapat berubah-ubah. Pengukuran secara otomatis menggunakan pendekatan deteksi objek pada pelampung dapat dilakukan untuk mempermudah proses monitoring ketinggian air.

Deteksi objek digunakan untuk mengetahui jarak pelampung dimana pelampung tersebut yang merepresentasikan ketinggian permukaan air. Penelitian ini menggunakan Algoritma Hough Circle Transform dikarenakan objek yang digunakan sebagai pelampung berbentuk lingkaran dimana Algoritma Hough Circle Transform dikhususkan untuk mendeteksi lingkaran. Selain itu Algoritma Hough Circle Transform dinilai tahan terhadap noise dan mampu mendeteksi lingkaran ketika sebagian lingkaran tidak terlihat. Penelitian [4] menggunakan Algoritma Hough Circle Transform untuk mendeteksi roda pada kendaraan dan hasilnya Algoritma Hough Circle Transform mampu mendeteksi semua roda dari 225 citra kendaraan dan beberapa deteksi non roda.

Penelitian [5] menggunakan Algoritma Hough Circle Transform untuk mendeteksi bola dengan variasi ukuran, jarak, dan intensitas cahaya. Hasilnya Algoritma Hough Circle Transform mampu mendeteksi bola dengan ukuran paling kecil 16.9 mm dan ukuran paling besar 95.8 mm, mampu mendeteksi bola dengan baik pada intensitas cahaya 117 lux dan 80 lux, dan mampu mendeteksi bola dengan baik pada jarak 40 cm sampai 140 cm. Pengujian yang dilakukan oleh [6] menggunakan sensor *ultrasonic* untuk mengukur ketinggian air. Ketinggian air dikategorikan menjadi tiga status yaitu Aman, Siaga 1, Siaga 2, dan Siaga 3. Status Aman akan ditunjukkan ketika permukaan air memiliki jarak 200 cm terhadap sensor sedangkan Status siaga ditunjukkan ketika permukaan air memiliki jarak 40 cm terhadap sensor. Penelitian [7] memanfaatkan konversi warna dari RGB ke HSV kemudian dengan menentukan nilai *threshold* untuk objek berwarna merah untuk mendapatkan nilai biner dari warna tersebut. Nilai ketinggian air didapatkan menggunakan rumus Regresi Polinomial. Hasil dari penelitian ini dengan melakukan 15 kali pengambilan data dari ketinggian 1 cm sampai 10 cm, terdapat kenaikan pixel secara polinomial pada ketinggian 1 cm sampai 7 cm, sedangkan ketinggian 8 cm sampai 10 cm menunjukkan penurunan *pixel*. Penelitian [8] menggunakan Persamaan Segitiga untuk mendapatkan jarak sebenarnya dari objek dengan memanfaatkan *focal length*, lebar objek sebenarnya dan lebar objek pada kamera. Hasil yang didapat dari penelitian ini Haar Cascade mampu mendeteksi objek dengan baik saat objek dalam keadaan bergerak, untuk perhitungan jarak sebenarnya setelah melakukan tiga kali percobaan pada dua kendaraan yang berbeda didapat persentase *error* 6,06% dan 20,76% pada kendaraan pertama dan 35,11% pada kendaraan kedua.

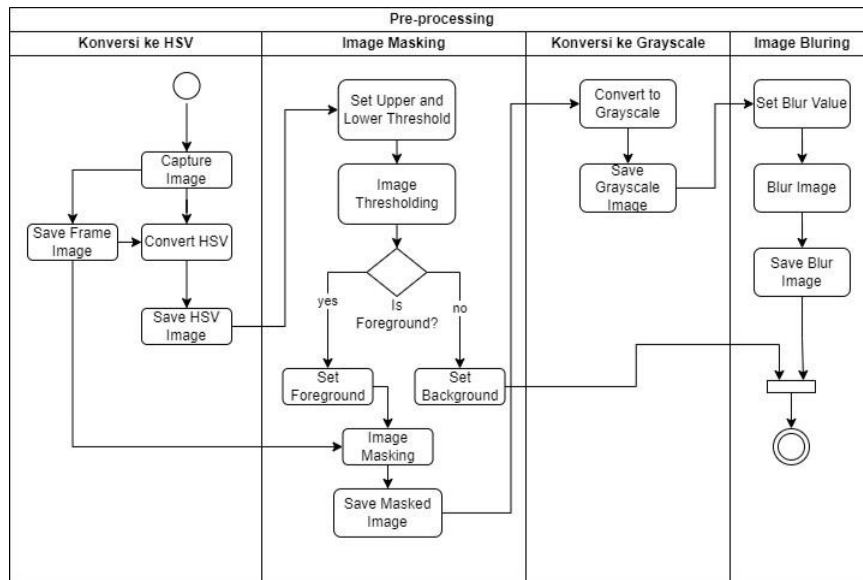
## 2. Research Method

Penelitian ini menggunakan pelampung berbentuk lingkaran sebagai objek yang dideteksi. Citra pelampung diambil menggunakan USB *webcam* kemudian melewati proses *masking* dan *bluring* sebelum proses deteksi objek menggunakan Algoritma Hough Circle Transform. Urutan Langkah-langkah penelitian ini dapat dilihat pada Gambar 1.



Gambar 1. Context Diagram Sistem

Sistem yang dirancang pada penelitian ini memiliki tiga komponen utama yaitu, Alat IoT, Server IoT, dan *Mobile Apps*. Alat IoT bertugas untuk mendapatkan dan mengirimkan data ketinggian air ke *database* Firebase. Server IoT bertugas untuk mengirim data report dan notifikasi ke *mobile apps* dan menyimpan data ketinggian air ke *MySQL database*. *Mobile Apps* bertugas untuk menampilkan data *report* dan data *real-time* dari ketinggian ke pengguna dan mengatur konfigurasi alat IoT.



Gambar 2. Alur PreProcessing

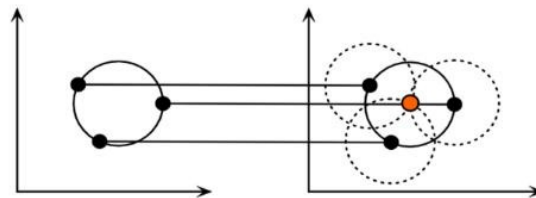
Proses *pre-processing* dimulai dengan proses konversi citra ke HSV kemudian dilakukan proses *masking* untuk memisahkan *background* dan *foreground*. Hasil *masking* akan di konversi ke *grayscale* dan di *blur* menggunakan *Gaussian blur*. Proses *pre-processing* ini dilakukan untuk mengambil ROI objek dan memperhalus citra untuk mengurangi *noise* saat proses deteksi lingkaran menggunakan *Hough Circle Transform*. Pengambilan ROI dilakukan untuk membantu

deteksi objek dengan menghilangkan bagian dari citra yang bukan bagian dari objek yang ingin dideteksi. Hal ini akan mempercepat proses deteksi dan mengurangi peluang terdeteksinya lingkaran yang bukan dari objek karena Hough Circle Transform bekerja dengan cara menggenerate lingkaran pada setiap tepi yang terdeteksi.

### 3. Literature Study

#### 3.1 Hough Circle Transform

Metode deteksi objek yang digunakan pada sistem ini adalah Hough Circle Transform, sedangkan metode untuk deteksi jarak yang digunakan adalah Persamaan Segitiga. Hough Circle Transform merupakan salah satu algoritma yang digunakan untuk mendeteksi lingkaran dalam sebuah gambar. Hough Circle Transform menggunakan proses voting yang menghitung probabilitas titik tepi pada sebuah lingkaran [9]. Hough Circle Transform menandai tepi lingkaran dengan radius  $R$  menggunakan titik, dan setiap titik membuat satu lingkaran dengan radius  $R$ . Titik tengah dari lingkaran didapat setelah menentukan titik intersepsi dari semua lingkaran yang di generate. Titik intersepsi yang didapat tersebut berubah menjadi titik tengah lingkaran yang memiliki radius  $R$  [10].

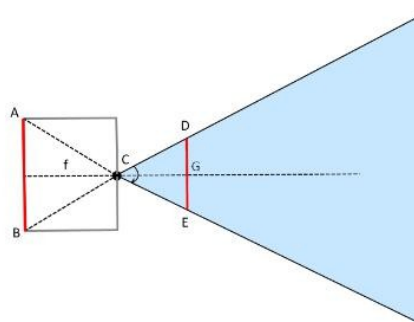


Gambar 3. Ilustrasi Hough Circle Transform

Gambar 3 merupakan ilustrasi Hough Circle Transform dimana setiap titik hitam pada tepi lingkaran dapat menggenerate lingkaran dengan radius  $R$ . Titik pusat sebenarnya dari lingkaran ditandai dengan titik merah yang dicari menggunakan Hough Accumulation Array.

#### 3.2 Persamaan Segitiga

Dua buah segitiga dapat dikatakan serupa jika memiliki besaran sudut yang bersesuaian sama besar dan sisi-sisi yang bersesuaian memiliki rasio atau perbandingan yang sama. Segitiga dapat dikatakan serupa jika memiliki panjang sisi yang berbeda namun setiap sudutnya persis [11].



Gambar 4. Persamaan Segitiga

Gambar 4 menunjukkan proyeksi gambar pada kamera pinhole. Dari hasil proyeksi gambar tersebut terbentuk dua segitiga yaitu ABC dan CDE. Menggunakan persamaan segitiga maka dapat ditulis.

$$AB/f = DE/CG$$

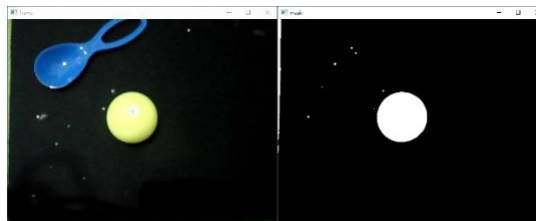
Nilai  $f$  merupakan *focal length* pada kamera,  $DE$  merupakan tinggi objek  $CG$  merupakan jarak objek ke kamera dan  $AB$  merupakan tinggi proyeksi. Rumus untuk mendapat jarak objek-kamera dapat dilihat pada Persamaan

$$CG = (f * DE) / AB$$

Sebagai contoh sebuah objek dengan jarak yang tidak diketahui direkam menggunakan kamera yang memiliki nilai *focal length* 123.135 mm dimana ukuran objek sebesar 38 mm Kamera mendeteksi objek tersebut dengan ukuran 118-pixel dikalikan dengan nilai 0.264 mengubah *pixel* ke satuan mm menjadi 31.194. Nilai  $f$  sebesar 123.135, nilai  $DE$  sebesar 38mm dan Nilai  $AB$  sebesar 31.220 mm. Berdasarkan Persamaan diatas maka didapat jarak objek terhadap kamera atau  $CG$  sebesar 149.876 mm.

### 3.3 Thresholding

Segmentasi citra merupakan salah satu tahapan inti dalam pengenalan objek dan klasifikasi pada *Computer Vision* [12]. Terdapat banyak teknik dalam proses segmentasi citra salah satunya adalah *Thresholding*. *Thresholding* digunakan untuk membagi citra menjadi *background* dan *foreground* [13]. *Thresholding* dilakukan dengan menentukan batas atas dan batas bawah pada nilai suatu warna pada citra, contohnya pada citra dengan *color space* HSV terdapat 3 batas bawah dan 3 batas atas untuk masing-masing *Hue*, *Saturation*, dan *Value* [14]. Hasil dari proses *thresholding* adalah sebuah citra binary.



Gambar 5. Hasil *Thresholding*

*Thresholding* bekerja dengan mengubah nilai *pixel* yang nilainya dibawah *threshold* ke nilai 0 dan yang nilainya diatas *threshold* ke nilai 1. Hasil dari proses ini adalah citra biner seperti Gambar 5 dimana *threshold* di konfigurasi ke warna bola sehingga objek lain seperti sendok berwarna biru tidak terlihat pada hasil *thresholding*.

### 3.3 Raspberry Pi

Raspberry Pi merupakan perangkat komputer *single-board* sebesar kartu kredit yang memiliki fitur yang sama dengan komputer pada umumnya [15]. Raspberry Pi memiliki *processor* ARM yang bekerja sebagai CPU dan GPU dimana CPU bertugas menangani *input*, dan menghasilkan *output* sedangkan GPU bertugas untuk menampilkan dan memproses grafis. GPIO merupakan komponen yang bisa ditemukan pada Raspberry Pi. *Generic Pin on Integrated Circuit* (GPIO) berfungsi untuk menerima input atau mengirimkan output ke perangkat yang terhubung [16]. USB yang merupakan *port* koneksi umum, *Audio Out* untuk koneksi perangkat audio, dan HDMI untuk koneksi perangkat *display*. Raspberry Pi memiliki *power connector* usb 5v yang dapat dihubungkan ke *power supply*. OS yang dijalankan oleh Raspberry Pi adalah OS Raspian yang disimpan pada kartu micro-SD [17].

### 3.3 Firebase

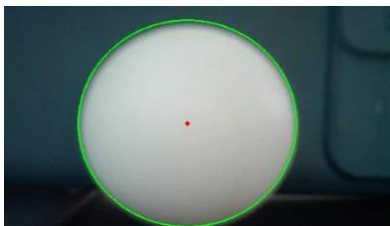
Firebase merupakan platform penyedia layanan dari google, firebase menyediakan banyak layanan seperti *real-time database*, *authentication*, *storage*, *cloud messaging*, *dashboard* dan lain-lain [18]. Pengembangan sistem monitoring sangat terbantu dengan adanya firebase *real-time database* dan *cloud messaging*. Firebase *real-time database* menyediakan akses secara *real-time* untuk beberapa pengembangan *mobile* seperti android dan iOS [19], dengan begitu, setiap perubahan data pada *database* maka setiap perangkat yang terhubung pada *database* tersebut mendapat update secara *real-time*.

#### 4. Result and Discussion

Sistem yang telah dibangun perlu diuji untuk mengetahui apakah metode pengukuran jarak mampu mengukur jarak dengan baik dan mengetahui apakah *mobile apps* mampu menampilkan data sesuai dengan jarak yang diukur oleh perangkat IoT.

##### 4.1. Deteksi Objek

Objek berbentuk lingkaran yang terdapat pada citra akan dideteksi menggunakan Algoritma Hough Circle Transform. Hasil dari deteksi ini adalah diameter lingkaran pada citra yang memiliki satuan *pixel* yang akan digunakan pada proses perhitungan *focal length* dan perhitungan jarak objek menggunakan Persamaan Segitiga. Indikator dari keberhasilan deteksi objek dapat dilihat dari lingkaran hijau yang mengelilingi objek.



Gambar 6. Hasil Deteksi Objek

Hasil deteksi objek yang berhasil dapat dilihat pada Gambar 6 dimana terdapat lingkaran hijau yang mengelilingi objek. Berdasarkan tiga percobaan yang dilakukan pada jarak yang berbeda, Hough Circle Transform mampu mendeteksi objek dari jarak 10 cm sampai 60 cm.

##### 4.2. Perhitungan Focal Length

Pengujian ini dilakukan untuk mengetahui apakah persamaan segitiga dapat menghitung focal length dengan variasi jarak dengan objek berdiameter 38 mm. Pengujian ini dilakukan dengan 6 variasi jarak yaitu 10 cm, 20 cm, 30 cm, 40 cm, 50 cm, 60 cm.

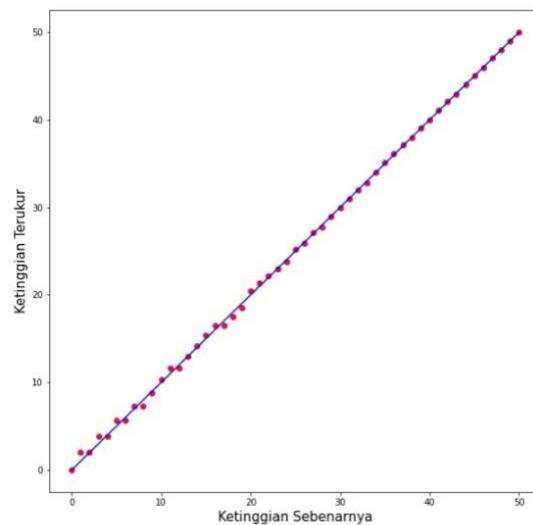
Tabel 1. Pengukuran *Focal Length*

Jarak	Focal Length
10	122.822368
30	124.702303
60	122.613487

Tabel 1 merupakan tabel data *focal length* yang didapat dengan variasi jarak objek terhadap kamera. Nilai *focal length* yang didapat memiliki satuan mm dengan nilai jarak dan diameter diubah ke satuan mm pada saat perhitungan *focal length*. Terlihat kalau nilai *focal length* tidak berubah secara signifikan walaupun nilai diameter dan jarak yang berubah. Hal ini terjadi karena setiap penambahan jarak juga terjadi pengurangan diameter dan nilai dari jarak dikali dengan diameter selalu mendekati 1.76.

##### 4.2. Pengujian Pengukuran Jarak

Jarak objek yang didapat dari alat IoT perlu diuji untuk mengetahui tingkat linearitas data. Data yang linear akan mempermudah proses kalibrasi pengukuran. Hasil pengukuran dapat dilihat pada Gambar 7.



Gambar 7. Hasil pengukuran jarak

Pengujian pengukuran jarak dilakukan dengan melakukan pengukuran objek dari jarak 10 sampai 60 cm menggunakan persamaan segitiga. Citra objek yang digunakan pada pengukuran diambil sebanyak 1000 sample dengan 20 sampel untuk setiap jarak pengukuran.

**Model Summary<sup>b</sup>**

Model	R	R Square	Adjusted R Square	Std. Error of the Estimate
1	1.000 <sup>a</sup>	.999	.999	.360

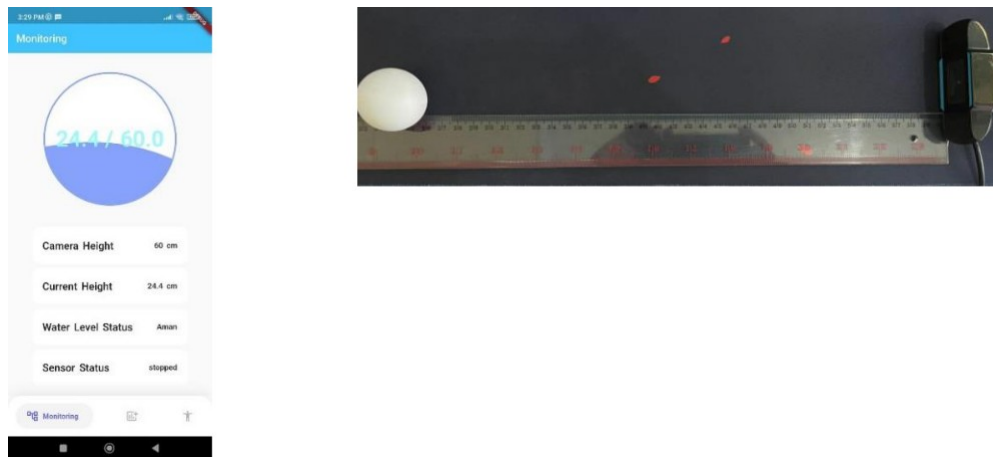
a. Predictors: (Constant), measured  
b. Dependent Variable: true

Gambar 8. Hasil Pengujian Linearitas

Nilai regresi dapat dilihat pada Gambar 8. Salah satu hasil dari analisa linear regression yang diperoleh menggunakan IBM SPSS yaitu tabel model *summary*. Tabel ini menunjukkan nilai  $R^2 = 1$  dan  $r = 0,999$  [20]. Berdasarkan persyaratan nilai korelasi ( $r$ ) yakni harus lebih besar dari 0,995, dengan nilai  $r$  yang diperoleh sebesar 0,999 maka dapat dinyatakan bahwa hasil linearitas valid dan terdapat korelasi antara jarak terukur dan jarak sebenarnya.

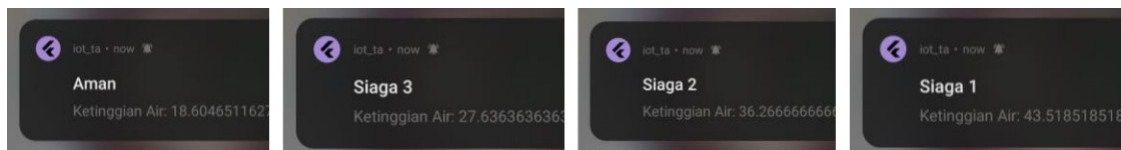
#### 4.2. Antarmuka Pengguna

Antarmuka yang dibuat perlu diuji untuk mengetahui apakah sudah sesuai dengan rancangan yang dibuat.



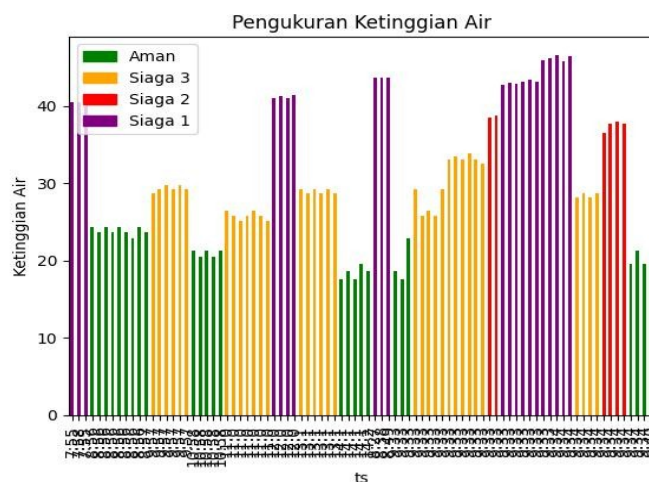
Gambar 9. Pengujian Aplikasi Mobile

Hasil pengukuran yang telah didapat akan dikirimkan ke aplikasi *mobile*. Tampilan pada *mobile apps* di sebelah kanan menunjukkan ketinggian air yang didapat pada perangkat IoT. TMA maksimum yang digunakan pada pengujian ini adalah 60 cm, sehingga ketinggian air dapat dihitung dengan mengurangi TMA maksimum dengan jarak yang didapat. Hasil dari pengukuran ketinggian air yang didapat pada pengujian ini adalah 24 cm dengan jarak antara objek terhadap kamera adalah 35.6 cm.



Gambar 10. Pengujian Notifikasi

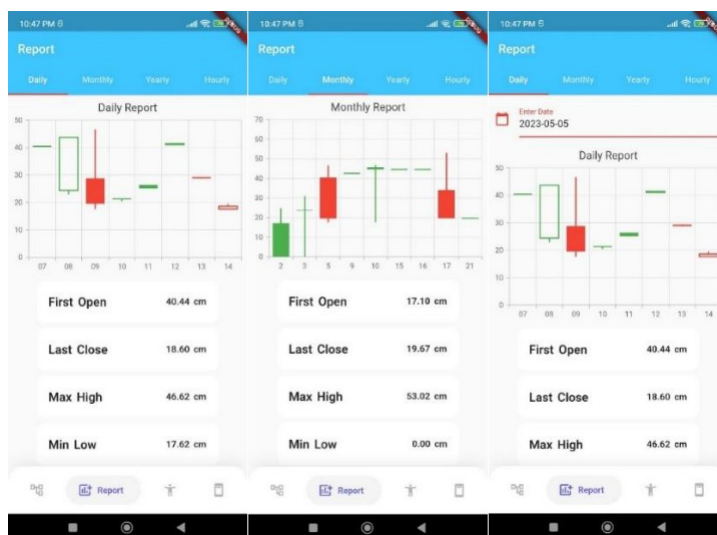
Hasil pengujian notifikasi yang dilakukan dapat dilihat pada Gambar 10. Notifikasi yang ditampilkan sudah sesuai dengan ketika ketinggian air yang diatur pada alat IoT. Notifikasi ditampilkan saat aplikasi sedang berjalan di *background*, sedangkan saat aplikasi sedang berjalan di *foreground*, data status ditampilkan pada halaman *water monitoring*.



Gambar 11. Grafik status ketinggian air



Gambar 11 merupakan tampilan grafik status ketinggian air yang didapat selama pengujian. Grafik tersebut menunjukkan waktu dan status ketinggian air pada waktu ketinggian air didapatkan. Warna ungu pada grafik menunjukkan status Siaga 1, warna merah menunjukkan status Siaga 2, warna oranye menunjukkan status Siaga 3, dan warna hijau menunjukkan status Aman.



Gambar 14. Grafik Report pada Mobile Apps

Gambar 14 menunjukkan tampilan dari halaman *reportDaily*, *report monthly*, dan *reportYearly* pada *mobile apps*. Data *report* ditampilkan dalam bentuk grafik dan pengguna dapat memilih interval dari report. Pengguna dapat memilih tanggal, bulan dan tahun yang diinginkan untuk menampilkan data yang didapat alat IoT pada interval yang disimpan pada MySQL database.

## 5. Conclusion

Berdasarkan hasil penelitian, kesimpulan yang didapat adalah sistem mampu data pengukuran monitoring ketinggian air menggunakan Algoritma Hough Circle Transform dan Persamaan Segitiga. Hasil pengukuran yang diperoleh akan langsung di *broadcast* ke setiap komponen sistem yang terhubung oleh Firebase. *Mobile apps* mampu menampilkan data ketinggian air dan mampu menampilkan notifikasi ketika ketinggian air mencapai batas tertentu. Selain itu, data ketinggian air yang diperoleh dapat dirangkum menjadi sebuah *report*. *Report* tersebut memiliki interval waktu harian, bulanan, dan tahunan ketika ketinggian air didapat. Data ketinggian air yang didapat juga memiliki *error rate* yang rendah dan memiliki hubungan yang linear dengan ketinggian air yang sebenarnya dengan tingkat linearitas sebesar 0.999 dan tingkat *error rate* sebesar 1.3757%. Hasil ini menyimpulkan kalau jarak objek terhadap kamera dapat dihitung menggunakan Persamaan Segitiga dengan bantuan Hough Circle Transform untuk sebagai algoritma deteksi objek dengan hasil yang cukup akurat.

Penelitian telah menunjukkan bahwa data ketinggian air yang diperoleh dari perangkat IoT dapat digunakan pada aplikasi *Mobile*. Perubahan ketinggian air dapat memberikan notifikasi ketika aplikasi sedang berjalan di *background* dengan data status ketinggian air dan ketinggian air saat notifikasi dikirimkan. Rancangan sistem yang mampu digunakan sebagai sistem monitoring ketinggian air serta mampu membuat rangkuman data ketinggian air pada interval harian, bulanan, dan tahunan

## References

- [1] K. S. Salamah and S. Anwar, "Rancang Bangun Sistem Pendeteksi Banjir Otomatis Berbasis Internet Of Things," *Jurnal Teknologi Elektro*, vol. 12, no. 1, p. 40, Jan. 2021, doi: 10.22441/jte.2021.v12i1.008.
- [2] D. Puspitasari, M. Thaifururrahman, and R. Ariyanto, "PENGEMBANGAN SISTEM

- PENDETEKSI BANJIR MENGGUNAKAN FUZZY DENGAN RASPBERRY PI (STUDI KASUS: KABUPATEN SAMPANG),” *Jurnal Teknologi Informasi dan Terapan*, vol. 4, no. 2, pp. 89–96, Apr. 2019, doi: 10.25047/jtit.v4i2.65.
- [3] N. Pratama, U. Darusalam, and N. D. Nathasia, “Perancangan Sistem Monitoring Ketinggian Air Sebagai Pendeteksi Banjir Berbasis IoT Menggunakan Sensor Ultrasonik,” *JURNAL MEDIA INFORMATIKA BUDIDARMA*, vol. 4, no. 1, p. 117, Jan. 2020, doi: 10.30865/mib.v4i1.1905.
- [4] S. D. Asri, D. Ramayanti, A. D. Putra, and Y. T. Utami, “Deteksi Roda Kendaraan Dengan Circle Hough Transform (CHT) dan Support Vector Machine (SVM),” *JURNAL TEKNOINFO*, vol. 16, pp. 427–434, 2022, doi: 10.33365/jti.v16i2.1952.
- [5] B. C. Wibowo, F. Nugraha, and A. P. Utomo, “Uji Deteksi Objek Bentuk Bola Dengan Menerapkan Metode Circular Hough Transform,” *JURNAL INFORMATIKA UPGRIS*, vol. 7, no. 1, 2021, doi: 10.26877/JIU.V7I1.8309.
- [6] Z. Zainuddin, A. Latif Arda, and A. Zulkifli Nusri, “SISTEM PERINGATAN DINI BANJIR,” *Jurnal Teknologi Informasi dan Komunikasi*, vol. 9, no. 2, pp. 167–173, 2019.
- [7] H. Fitri and dan Ivan Finiel Hotmartua Bagariang, “Pemanfaatan ESP32-CAM Untuk Mengukur Ketinggian Air Menggunakan Metode Image Processing,” *Seminar Nasional Terapan Riset Inovatif (SENTRINOV) Ke-6 ISAS Publishing Series: Engineering and Science*, vol. 6, no. 1, 2020, Accessed: Apr. 03, 2023. [Online]. Available: <https://proceeding.isas.or.id/index.php/sentrinov/article/view/547>
- [8] Axel Davlin Yusyahnur, Agus Virgon, and Umar Ali Ahmad, “Pengukuran Jarak Kendaraan Dengan Metode Haar Cascade Measurement Of Vehicle Distance With Haar Cascade Method Using Opencv,” *e-Proceeding of Engineering*, vol. 9, pp. 1145–1158, 2022.
- [9] R. J. Al Kautsar, F. Utaminigrum, and A. S. Budi, “Helmet Monitoring System using Hough Circle and HOG based on KNN,” *Lontar Komputer : Jurnal Ilmiah Teknologi Informasi*, vol. 12, no. 1, p. 13, Mar. 2021, doi: 10.24843/lkjiti.2021.v12.i01.p02.
- [10] A. H. Rodzi, Z. M. Zin, and N. Ibrahim, “Eye Gaze Detection Using Hough Circle Transform,” *International Journal of Applied Engineering Research*, vol. 12, no. 24, pp. 14452–14459, 2017, [Online]. Available: <http://www.ripublication.com>
- [11] A. Setiawan, H. Hadiyanto, and C. E. Widodo, “Distance Estimation Between Camera and Shrimp Underwater Using Euclidian Distance and Triangles Similarity Algorithm,” *Ingenierie des Systemes d’Information*, vol. 27, no. 5, pp. 717–724, Oct. 2022, doi: 10.18280/isi.270504.
- [12] T. Lei, X. Jia, Y. Zhang, S. Liu, H. Meng, and A. K. Nandi, “Superpixel-Based Fast Fuzzy C-Means Clustering for Color Image Segmentation,” *IEEE Transactions on Fuzzy Systems*, vol. 27, no. 9, pp. 1753–1766, Sep. 2019, doi: 10.1109/TFUZZ.2018.2889018.
- [13] E. A. Pambudi, E. S. Wijaya, and A. Fauzan, “IMPROVED SAUVOLA THRESHOLD FOR BACKGROUND SUBTRACTION ON MOVING OBJECT DETECTION,” *International Journal of Software Engineering and Computer Systems*, vol. 5, no. 2, pp. 78–89, Aug. 2019, doi: 10.15282/ijsecs.5.2.2019.6.0062.
- [14] Y. Ito, C. Premachandra, S. Sumathipala, H. W. H. Premachandra, and B. S. Sudantha, “Tactile Paving Detection by Dynamic Thresholding Based on HSV Space Analysis for Developing a Walking Support System,” *IEEE Access*, vol. 9, pp. 20358–20367, 2021, doi: 10.1109/ACCESS.2021.3055342.
- [15] I. N. Piarsa, I. W. P. Agustiana Putra, and K. Suar Wibawa, “Sistem Pendeteksi Kebakaran Menggunakan Raspberry Pi Berbasis Android,” *Jurnal Merpati*, vol. 6, no. 3, p. 167, 2018, doi: 10.24843/jim.2018.v06.i03.p03.
- [16] I. Gusti, P. Sena, S. Adnyana, I. Nyoman Piarsa, and K. S. Wibawa, “Internet of Things: Control and Monitoring System of Chicken Eggs Incubator Using Raspberry Pi,” *International Journal of Internet of Things*, vol. 7, no. 1, pp. 16–21, 2018, doi: 10.5923/j.ijit.20180701.03.
- [17] G. Widya Dharma, I. Nyoman Piarsa, and I. Made Agus Dwi Suarjaya, “Kontrol Kunci Pintu Rumah Menggunakan Raspberry Pi Berbasis Android,” *Jurnal Merpati*, vol. 6, no. DESEMBER, 2018.
- [18] S. Sarkar, S. Gayen, and S. Bilgaiyan, “Android based Home Security Systems using Internet of Things (IoT) and Firebase,” *2018 International Conference on Inventive Research in Computing Applications (ICIRCA)*, no. 1, pp. 102–105, 2018, doi: 10.1109/ICIRCA.2018.8597197.
- [19] Q. Hassan, “Internet of Things A to Z: Technologies and Applications,” 2018. [Online]. Available: <https://www.researchgate.net/publication/322538736>
- [20] M. Y. Suryani, A. Paramita, H. Susilo, and I. K. Maharsih, “Analisis Penentuan Kadar Besi (Fe) dalam Air Limbah Tambang Batu Bara Menggunakan Spektrofotometer UV-Vis,” *JOURNAL OF LABORATORY ISSN*, vol. 5, no. 1, p. 4887, 2022.