JITEL (Jurnal Ilmiah Telekomunikasi, Elektronika, dan Listrik Tenaga) p-ISSN: 2774-7972
Vol. 2, No. 1, Maret 2022, pp. 11-24 e-ISSN: 2775-6696
DOI: https://doi.org/10.35313/jitel.v2.i1.2022.11-24

Aplikasi pengukur kecepatan bola pada video pertandingan tenis meja menggunakan OpenCV dan Python

Eril Mozef[1], Siti Kurniati[2*]
[1,2]Jurusan Teknik Elektro, Politeknik Negeri Bandung
Jalan Gegerkalong Hilir, Ciwaruga, Kecamatan Parongpong, Kabupaten Bandung Barat, Indonesia
[1]erilmozef@gmail.com, [2*]siti.kurniati.tkom18@polban.ac.id

ABSTRAK

Dalam cabang olahraga tenis meja, seringkali kita ingin menganalisa karakteristik bola (kecepatan, spin, dan lain-lain) secara kuantitatif. Hal ini dikarenakan selama ini, analisa dilakukan secara kualitatif melalui pengamatan visual saja. Dengan adanya sarana analisa karakterisik bola ini maka kemampuan lawan maupun pemain itu sendiri dapat dievaluasi guna mengatur strategi pelatihan yang baik. Terkait dengan itu maka suatu aplikasi berbasis *software* pengukur kecepatan bola dari sebuah video pertandingan tenis meja dikembangkan pada penelitian ini. Pada aplikasi ini, pengguna dapat membuka sebuah file video pertandingan tenis meja, mengamati lintasan bola yang ingin diukur, menempatkan dua kotak pengamatan kemunculan bola di layar, memainkan kembali video, dan dapat mengetahui nilai kecepatan yang ditampilkan di layar. Aplikasi ini menggunakan *platform* OpenCV dan Python. Penelitian ini telah mampu mengukur kecepatan bola pada rentang 6 km/jam hingga 295 km/jam dengan beberapa asumsi antara lain segmen lintasan yang dipilih pendek, lurus, dan berada di tengah lintasan sehingga faktor percepatan dapat diabaikan dalam perhitungan. Walaupun hasil yang didapat masih belum terlalu presisi, namun aplikasi ini sudah dapat memberikan informasi berharga tentang seberapa cepat bola dimainkan dalam suatu pertandingan tenis meja.

Kata kunci: tenis meja, bola, pengukuran kecepatan, OpenCV, Python

ABSTRACT

In table tennis, we often want to analyze the characteristics of the ball (speed, spin, etc.) quantitatively. This is because so far, the analysis has been done qualitatively through visual observation only. With this means of analyzing the characteristics of the ball, the ability of the opponent and the players themselves can be evaluated in order to arrange a good training strategy. Related to that, a software-based application for measuring ball speed from a ping pong video game was developed in this study. In this application, users can open a table tennis match video file, observe the trajectory of the ball to be measured, place two observation boxes for the appearance of the ball on the screen, play back the video, and can find out the speed value displayed on the screen. This application uses the OpenCV and Python platforms. This research has been able to measure the speed of the ball in the range of 6 km / h to 295 km / h with several assumptions, including the selected trajectory segment is short, straight, and is in the middle of the trajectory so that the acceleration factor can be neglected in the calculation. Although the results obtained are still not very precise, this application can provide valuable information about how fast the ball is played in a table tennis match.

Keywords: *table tennis, ball, speed measurement, OpenCV, Python*

1. PENDAHULUAN

Pada pertandingan tenis meja, kecepatan pukulan bola merupakan salah satu faktor penting yang harus dimiliki oleh pemain. Namun, dalam menentukan strategi permainan diperlukan data kuantitatif seberapa cepat bola yang dipukul. Pelatih dapat memberikan masukan ke atlet yang sedang dilatihnya, agar dapat meningkatkan kekuatan pukulannya dengan membandingkannya dengan kekuatan pemain lawan serta mengatur strategi pertandingan. Beberapa produk pengukur kecepatan bola seperti untuk mengukur kecepatan bola tenis lapangan, *baseball*, dan lain-lain telah ada di pasaran. Contohnya *speed gun* untuk mengukur kecepatan bola tenis saat di-*servis* [1] dan *radar gun* untuk mengukur kecepatan melintas bola tenis [2], [3]. Namun, produk ini memiliki keterbatasan hanya dapat

Naskah diterima tanggal 29 Juli 2021, disetujui tanggal 31Januari 2022

*E-mail korespondensi

digunakan di lapangan (*onsite*) dan tidak dapat digunakan untuk menganalisa permainan lawan dalam sebuah pertandingan secara *offsite*.

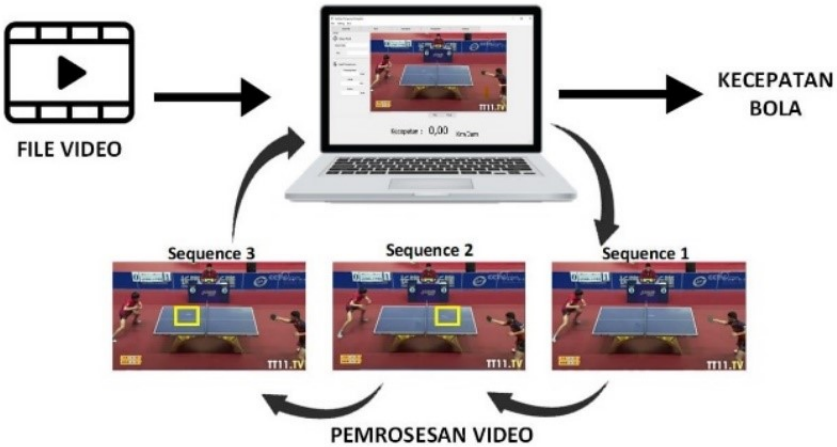
Selain itu terdapat juga beberapa artikel yang dijadikan sebagai referensi pada penelitian ini. Seperti pada penelitian [4] dimana sistem *object tracking* diterapkan dengan metode pelacakan *adaptive particle filter* yang berfungsi untuk menghilangkan noise yang dihasilkan ketika melakukan deteksi lingkaran bola. Kemudian pendeteksi dan pengukur jarak bola pada permainan sepak bola dengan metode *optical flow* dan *template matching* telah dikembangkan penelitian [5]. Kombinasi kedua metode ini digunakan untuk memperkuat pelacakan dan memperkecil galat yang dihasilkan. Algoritma pelacakan lainnya seperti *color filtering* HSV (*hue, saturation, value*) pada *robot soccer humanoid* dan sistem pendeteksi jarak telah diterapkan penelitian [6] dan [7]. Hasilnya metode ini mampu mentoleransi perubahan intensitas cahaya dalam mendeteksi warna benda. Selanjutnya pengukur jarak bola terhadap kamera dengan metode *find contour* telah dikembangkan pada penelitian [8]. Dengan metode ini objek bola dapat dideteksi berdasarkan jarak dan posisinya terhadap kamera. Kemudian sistem *tracking* bola dengan metode Kalman *filter*, Hough Circle Transform, *blob detection*, dan Camshift juga telah dikembangkan pada penelitian [9], [10]. Dari berbagai metode *image processing* di atas, dapat disimpulkan bahwa pendeteksian objek khususnya dalam hal ini bola memerlukan algoritma tertentu untuk menghasilkan tingkat akurasi maksimal.

Penelitian ini bertujuan untuk mengembangkan sebuah *software* aplikasi yang mampu mengukur kecepatan bola pada pertandingan tenis meja secara *offsite* (di luar lapangan). Berbeda dengan penelitian sebelumnya di atas, pada penelitian ini sistem dikembangkan untuk aplikasi pendeteksian kecepatan bola. Selain itu penelitian ini difokuskan untuk pengembangan aplikasi yang digunakan untuk kebutuhan analisis pertandingan tenis meja. Dengan aplikasi ini maka dapat dilakukan analisis bola yang melintas diantara dua daerah pengamatan ROI (*Region of Interest*) menggunakan teknik pengolahan citra berbasis OpenCV dan Phyton.

2. METODE PENELITIAN

2.1 Rancangan Aplikasi

Pada aplikasi ini, pengguna dapat melakukan beberapa hal seperti membuka sebuah *file* video dalam format MP4 (30 fps) pertandingan tenis meja, mengamati lintasan bola yang ingin diukur, menempatkan dua kotak pengamatan kemunculan bola di layar, memainkan kembali video, dan melihat nilai kecepatan yang didapatkan di layar. Adapun ilustrasi lengkap dari aplikasi ini ditunjukkan seperti pada Gambar 1.



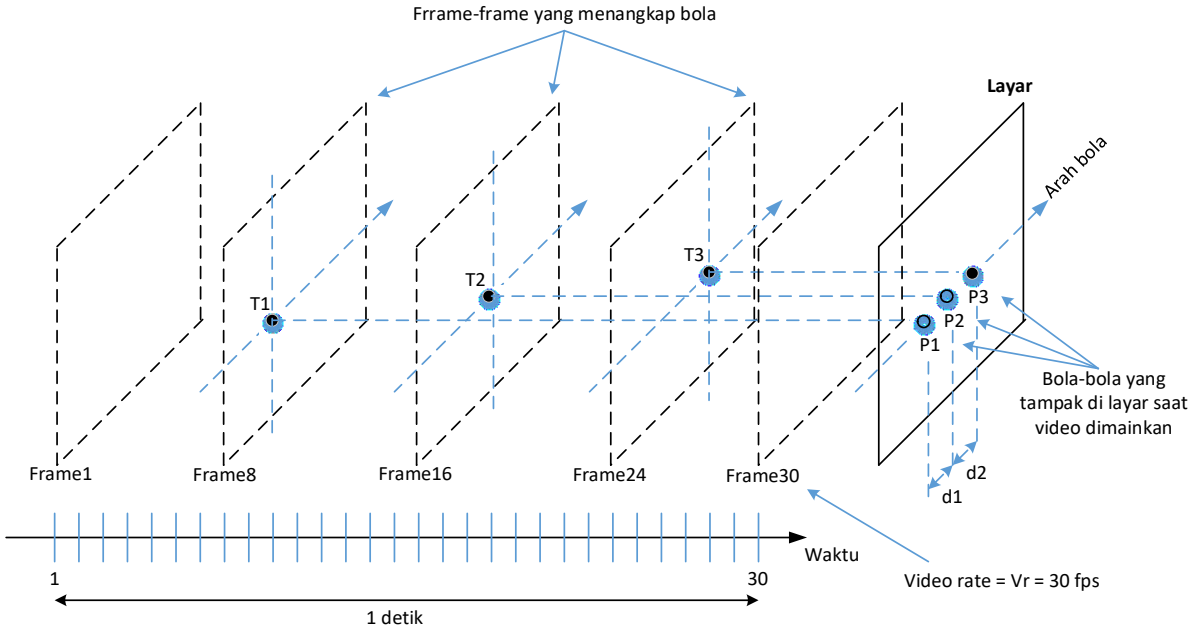
Gambar 1. Ilustrasi aplikasi yang dikembangkan

Dikarenakan dalam pengembangan aplikasi ini diterapkan beberapa batasan, aplikasi ini untuk sementara belum dapat dibandingkan hasilnya dengan produk pengukur jarak *handheld* profesional ataupun aplikasi *software* komersial. Aplikasi ini nantinya hanya akan dipergunakan untuk membandingkan kecepatan bola dalam video yang sama atau video yang berbeda dengan format yang sama. Adapun beberapa batasan yang diberlakukan antara lain:

1. Perhitungan dilakukan dalam bidang 2 dimensi dengan sumbu XY dikarenakan video yang digunakan video 2 dimensi.
2. Gambar yang diolah tampak samping agar mendapatkan sebanyak mungkin gerakan dalam sumbu XY dan meminimalisir gerakan dalam sumbu Z.
3. Segmen lintasan yang dipilih pendek untuk memperkecil efek parabola bola.
4. Segmen lintasan yang dipilih berada di tengah lintasan dan lurus agar bisa diasumsikan kecepatan awal sama dengan akhir (agar percepatan sama dengan nol) dan perhitungan dapat menggunakan rumus GLB (Gerak Lurus Beraturan).
5. Kecepatan video menggunakan standar kecepatan video normal yang paling rendah yaitu 30 fps sehingga untuk kecepatan tinggi bola hanya dapat ditangkap oleh beberapa *frame* saja namun metoda yang digunakan masih dapat mentolerir hal ini.
6. Aplikasi menggunakan *platform* OpenCV dan Python.

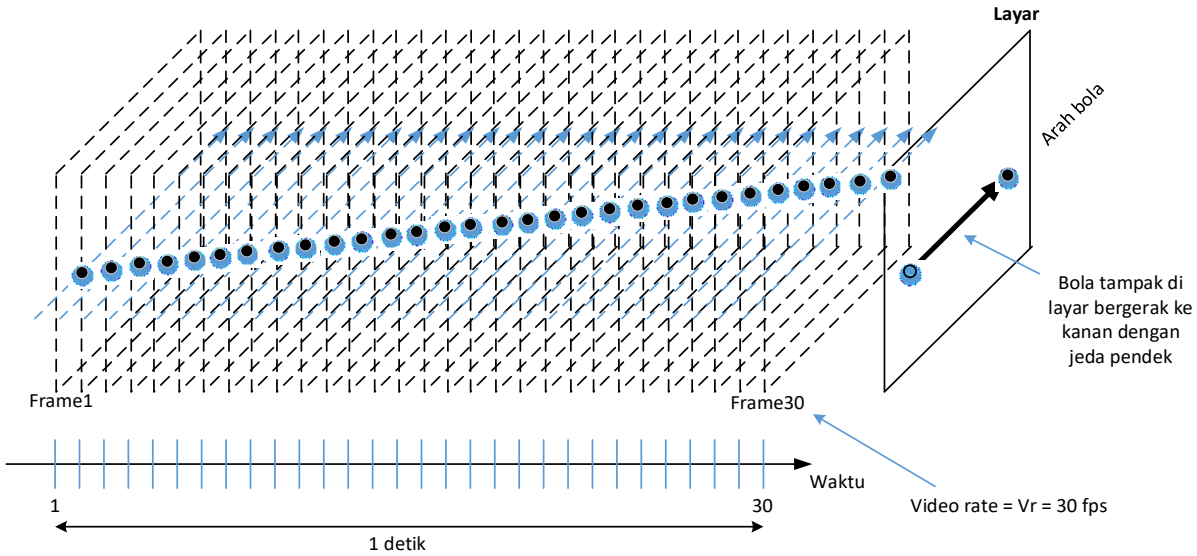
2.2 Perhitungan Waktu Bola

Pada sebuah video 30 fps, sebuah bola melintas secara horizontal di layar seperti diilustrasikan Gambar 2. Kecepatan bola menyebabkan bola hanya dapat ditangkap oleh beberapa *frame* saja yaitu *frame* 8, *frame* 16, dan *frame* 24 sehingga bola muncul berurutan dari kiri ke kanan seperti tampak pada layar. Gambar yang ditampilkan di layar memperlihatkan pergerakan bola dengan banyak jeda *pixel* kosong. Gambar 3 adalah ilustrasi dimana bola dengan waktu melintas di layar lebih besar atau sama dengan 1 detik. Dengan demikian, seluruh *frame* akan menangkap bola dan bola akan terlihat jelas dan tidak putus-putus pada layar. Gambar 4 adalah ilustrasi dimana bola dengan waktu melintas di layar lebih kecil atau sama dengan 1/Vr dalam hal ini 1/30 detik. Dengan demikian, hanya ada 2 *frame* yang menangkap bola dan bola akan terlihat tidak jelas dan terputus-putus pada layar. Contoh pada Gambar 4, indeks *frame* yang menangkap bola adalah 6 dan 7.

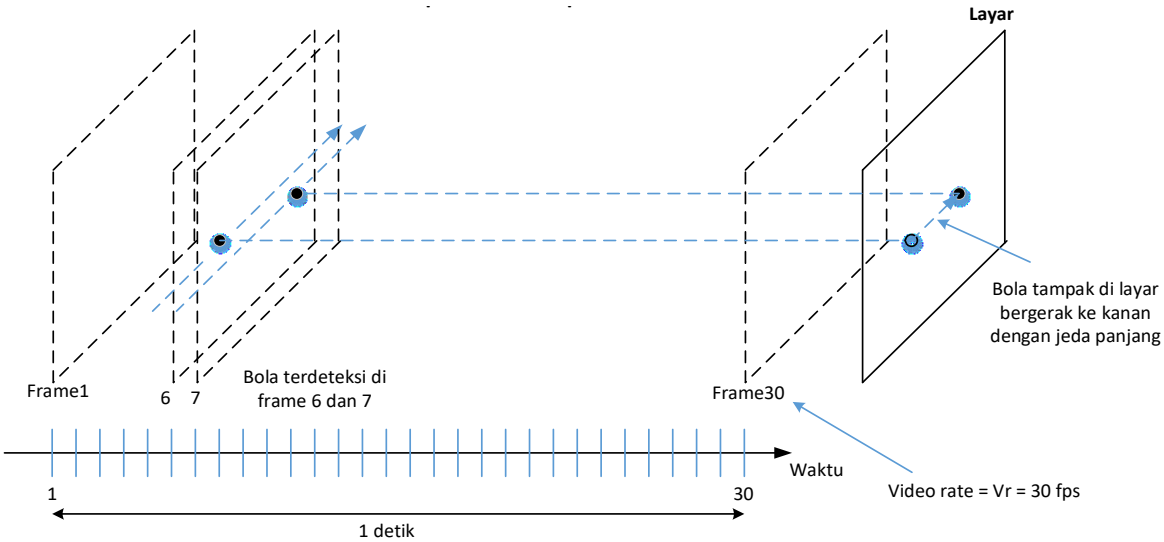


T1: waktu pertama frame berhasil menangkap bola. Frame ini frame ke-8 dari 30 frame.
T2: waktu kedua frame berhasil menangkap bola. Frame ini frame ke-16 dari 30 frame.
T3: waktu ketiga frame berhasil menangkap bola. Frame ini frame ke-24 dari 30 frame.
P1: posisi bola di layar yang tertangkap frame pada T1.
P2: posisi bola di layar yang tertangkap frame pada T2.
P3: posisi bola di layar yang tertangkap frame pada T3.
d1: jarak P1-P2
d2: jarak P2-P3

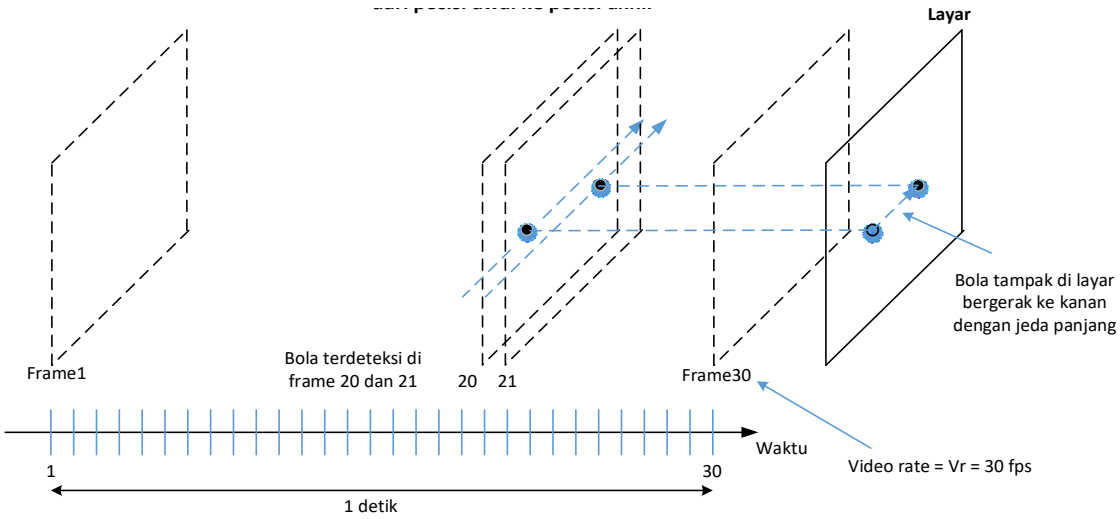
Gambar 2. Ilustrasi prinsip perhitungan waktu dan jarak bola pingpong berdasarkan kemunculan dan posisi di layar



Gambar 3. Ilustrasi kondisi bola bergerak paling lambat dengan waktu 1 detik

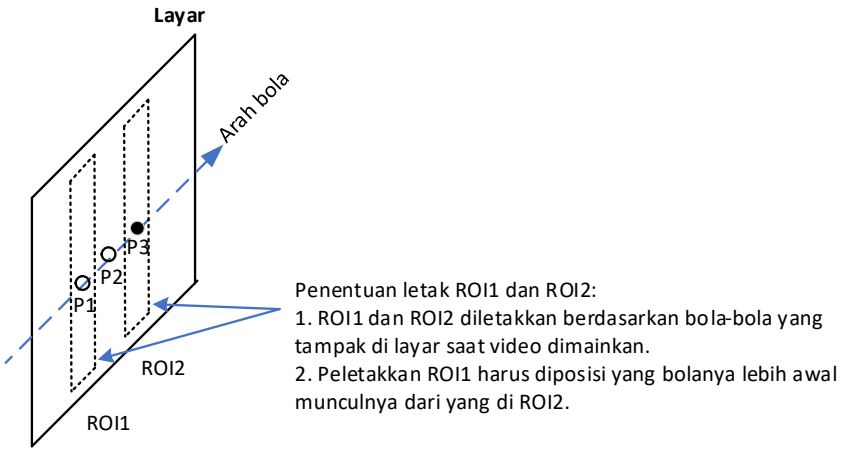


Gambar 4. Ilustrasi bola bergerak dengan waktu 1/30 detik terdeteksi pada frame ke-6 dan ke-7



Gambar 5. Ilustrasi bola bergerak dengan waktu 1/30 detik terdeteksi pada frame ke-20 dan ke-21

Gambar 5 adalah ilustrasi dimana bola dengan waktu melintas di layar lebih kecil atau sama dengan 1/Vr dalam hal ini 1/30 detik untuk *sampling frame* berada pada indeks yang berbeda dengan sebelumnya yaitu pada indeks 20 dan 21. Untuk dapat menghitung waktu, maka kita harus memilih segmen gerak bola bagian mana yang ingin diukur waktunya.



Gambar 6. Ilustrasi pergerakan bola

Pada Gambar 6, diperlihatkan bahwa kita dapat memilih daerah pengamatan pada layar dengan cara menempatkan 2 kotak ROI1 (kiri) dan ROI2 (kanan). Dengan menggunakan teknik pengolahan citra yang akan dibahas pada bagian berikutnya, bola pada ROI1 akan terdeteksi posisi (X1,Y1) dan indeks *frame*-nya. Begitu pula bola pada ROI2 akan terdeteksi posisi (X2,Y2) dan indeks *frame*-nya. Dengan mengetahui indeks *frame* masing-masing maka kita dapat menghitung jumlah *frame* yang berada diantaranya dengan cara mengurangi indeks *frame* ke-2 dan ke-1. Dengan mengetahui kecepatan video yang digunakan, maka kita dapat mengkonversi jumlah *frame* ini menjadi satuan waktu dan waktu tersebut adalah waktu bola melintas dari titik P1P2 pada ROI1 dan ROI2. Untuk kecepatan video (Vr) 30 fps yang digunakan artinya dalam 1 detik akan terdapat 30 *frame*. Dengan demikian waktu yang dibutuhkan bola melintas dari titik P1P2 untuk jumlah *frame* (Jmf) adalah sebagai berikut.

$$Waktu\ P1P2 = \frac{Jmf}{Vr}\ (detik) \tag{1}$$

2.3 Perhitungan Jarak Bola

Pada saat menempatkan ROI1, selain kita mendapatkan indeks *frame* maka kita akan mendapatkan juga koordinat P1(X1,Y1). Begitu pula, pada ROI2 kita akan mendapatkan koordinat P2(X2,Y2). Dari sini kita dapat menghitung jarak P1-P2 sebagai berikut.

$$Jarak\ P1P2 = Jb_layar_px = \sqrt{(x_{2b} - x_{1b})^2 + (y_{2b} - y_{1b})^2} \tag{2}$$

Adapun jarak bola di layar dalam satuan cm dapat dihitung sebagai berikut.

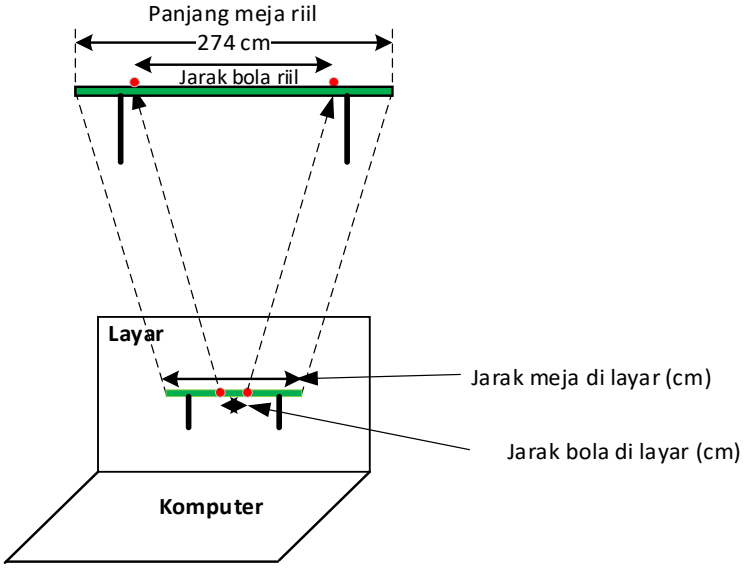
$$Jb_layar_cm = Jb_layar_px * Jp_layar_cm \tag{3}$$

dengan *Jb_layar_px* adalah jarak bola di layar dalam satuan pixel yang didapat dari persamaan (2) dan *Jp_layar_cm* adalah jarak 1 pixel dalam cm. Dalam sistem video jarak ini sudah diketahui sebesar 1920 pixels / 20 inch atau sama dengan 0,0264583333 cm. Jika persamaan (2) disubtitusikan ke dalam persamaan (3) maka jarak dalam cm ditunjukkan seperti persamaan (4).

$$Jb_layar_cm = \sqrt{(x_{b2} - x_{b1})^2 + (y_{b2} - y_{b1})^2} * Jp_layar_cm \quad (4)$$

Jp_layar_cm walaupun sudah diketahui besarnya yaitu 0,0264583333 cm, namun variabel ini kita biarkan tetap ada dalam persamaan karena nantinya akan hilang dengan sendirinya pada penyederhanaan persamaan berikutnya.

2.4 Perhitungan Jarak Bola Riil



Gambar 7. Jarak bola riil di meja fungsi dari jarak bola di layar

Dengan mengetahui standar ukuran panjang meja pingpong adalah 274 cm dan lebar 152,5 cm serta dengan prinsip perbandingan seperti yang ditunjukkan pada ilustrasi Gambar 7, maka jarak bola riil di meja dapat ditentukan sebagai berikut.

$$Jb_riil_cm = \frac{Panjang_meja_riil_cm}{Panjang_meja_layar_cm} * Jb_layar_cm \quad (5)$$

dengan *Panjang_meja_riil_cm* adalah panjang meja tenis sebenarnya yaitu 274 cm, *Jb_layar_cm* sudah diketahui dan didapat dari persamaan (4), serta *Panjang_meja_layar_cm* didapat dengan menentukan 2 titik meja di layar oleh pengguna. Adapun panjang meja riil dapat ditentukan sebagai berikut.

$$Panjang_meja_px = \sqrt{(x_{m2} - x_{m1})^2 + (y_{m2} - y_{m1})^2} \quad (6)$$

$$Panjang_meja_layar_cm = \sqrt{(x_{m2} - x_{m1})^2 + (y_{m2} - y_{m1})^2} * Jp_{layar_{cm}} \quad (7)$$

$$Jb_riil_cm = \frac{274 * \sqrt{(x_{b2} - x_{b1})^2 + (y_{b2} - y_{b1})^2} * Jp_layar_cm}{\sqrt{(x_{m2} - x_{m1})^2 + (y_{m2} - y_{m1})^2} * Jp_layar_cm} \quad (8)$$

Disini terlihat bahwa variabel *Jp_layar_cm* saling menghilangkan, sehingga dihasilkan persamaan lebih sederhana.

$$Jb_riil_cm = 274 * \sqrt{\frac{(x_{2b} - x_{1b})^2 + (y_{2b} - y_{1b})^2}{(x_{2m} - x_{1m})^2 + (y_{2m} - y_{1m})^2}} \quad (9)$$

2.5 Perhitungan Kecepatan Bola

Setelah persamaan jarak dan waktu telah didapatkan maka kecepatan dalam satuan cm/detik dapat dihitung dengan persamaan berikut.

$$V = \frac{Jb_riil_cm}{Waktu\ P1P2}\ (cm/dtk) \tag{10}$$

$$V = \frac{274 * \sqrt{\frac{(x_{2b} - x_{1b})^2 + (y_{2b} - y_{1b})^2}{(x_{2m} - x_{1m})^2 + (y_{2m} - y_{1m})^2}}}{\frac{Jmf}{Vr}}\ (cm/dtk) \tag{11}$$

$$V = \frac{274 * Vr}{Jmf} * \sqrt{\frac{(x_{2b} - x_{1b})^2 + (y_{2b} - y_{1b})^2}{(x_{2m} - x_{1m})^2 + (y_{2m} - y_{1m})^2}}(cm/dtk) \tag{12}$$

Adapun kecepatan bola dalam km/jam dapat dihitung sebagai berikut.

$$V = \frac{274 * Vr}{Jmf} * \sqrt{\frac{(x_{2b} - x_{1b})^2 + (y_{2b} - y_{1b})^2}{(x_{2m} - x_{1m})^2 + (y_{2m} - y_{1m})^2}}\frac{10^{-5}km}{\frac{1}{3600}j} \tag{13}$$

$$V = \frac{274 * 3600 * 10^{-5} * Vr}{Jmf} * \sqrt{\frac{(x_{2b} - x_{1b})^2 + (y_{2b} - y_{1b})^2}{(x_{2m} - x_{1m})^2 + (y_{2m} - y_{1m})^2}}(km/j) \tag{14}$$

$$V = \frac{9{,}864 * Vr}{Jmf} * \sqrt{\frac{(x_{2b} - x_{1b})^2 + (y_{2b} - y_{1b})^2}{(x_{2m} - x_{1m})^2 + (y_{2m} - y_{1m})^2}}(km/j) \tag{15}$$

Mengacu pada persamaan (15) nilai 9,864 adalah konstanta kecepatan pada panjang meja 274 cm dan kecepatan dalam km/jam. Lalu Vr merupakan kecepatan video (fps) dan Jmf adalah jumlah *frame* antara dua *frame* yang dapat menangkap bola. Kemudian X_{1m}, Y_{1m}, X_{2m}, Y_{2m} adalah koordinat meja di layar dan X_{1b}, Y_{1b}, X_{2b}, Y_{2b} adalah koordinat bola dilayar.

2.6 Algoritma Mencari Koordinat Meja, Bola, dan Jumlah *Pixel*

Rumus kecepatan pada persamaan (12) maupun (15) dapat diimplementasi cukup dengan mencari koordinat meja di layar, koordinat bola dilayar, dan nilai Jmf menggunakan program. Koordinat meja di layar dapat dicari dengan meminta pengguna menentukan 2 titik pada ujung-ujung meja di layar. Koordinat bola dilayar dan nilai Jmf dapat dicari dengan meminta pengguna untuk menentukan 2 buah ROI pada kemunculan bola-bola di layar saat pengamatan awal. Koordinat meja dapat dicari dengan menentukan 2 buah titik pada ujung-ujung tengah bola menggunakan perangkat *mouse* pada komputer. Koordinat ini didapat dengan menggunakan fungsi mendapatkan koordinat pixel pada video yang telah dimasukan. Ketika ujung-ujung meja dipilih menggunakan *mouse* komputer, maka koordinat *pixel* meja akan diambil. Koordinat ini kemudian disimpan dalam suatu variabel *self.koord1* sebagai X_{1m},Y_{1m} dan *self.koord2* sebagai X_{2m}, ,Y_{2m}. Koordinat bola dicari dengan cara mendeteksi bola yang masuk ke dalam dua lintasan (ROI) yang telah dibuat dengan menggunakan algoritma *contour detection* untuk proses pendeteksiannya. Bola yang terdeteksi memasuki kedua lintasan akan dibuat garis tepi (*contour*) di sekeliling bola. Kemudian koordinat *pixel contour* bola akan dijadikan sebagai koordinat bola. Ketika koordinat *pixel contour* yaitu *self.kbx* dan *self.kby* memasuki daerah lintasan ke-1 yaitu x1, x2, y1 dan y2, maka akan diambil koordinat *pixel contour* tersebut dan disimpan dalam variabel *self.koorb1* sebagai koordinat bola X_{1b}, Y_{1b}.. Lalu ketika koordinat *pixel contour self.kbx* dan

self.kby terdeteksi pada daerah lintasan ke-2 yaitu xr1, xr2, yr1 dan yr2, maka diambil koordinat *pixel contour* bola tersebut dan disimpan dalam variabel *self.koorb2* sebagai koordinat $X_{2m}Y_{2m}$.

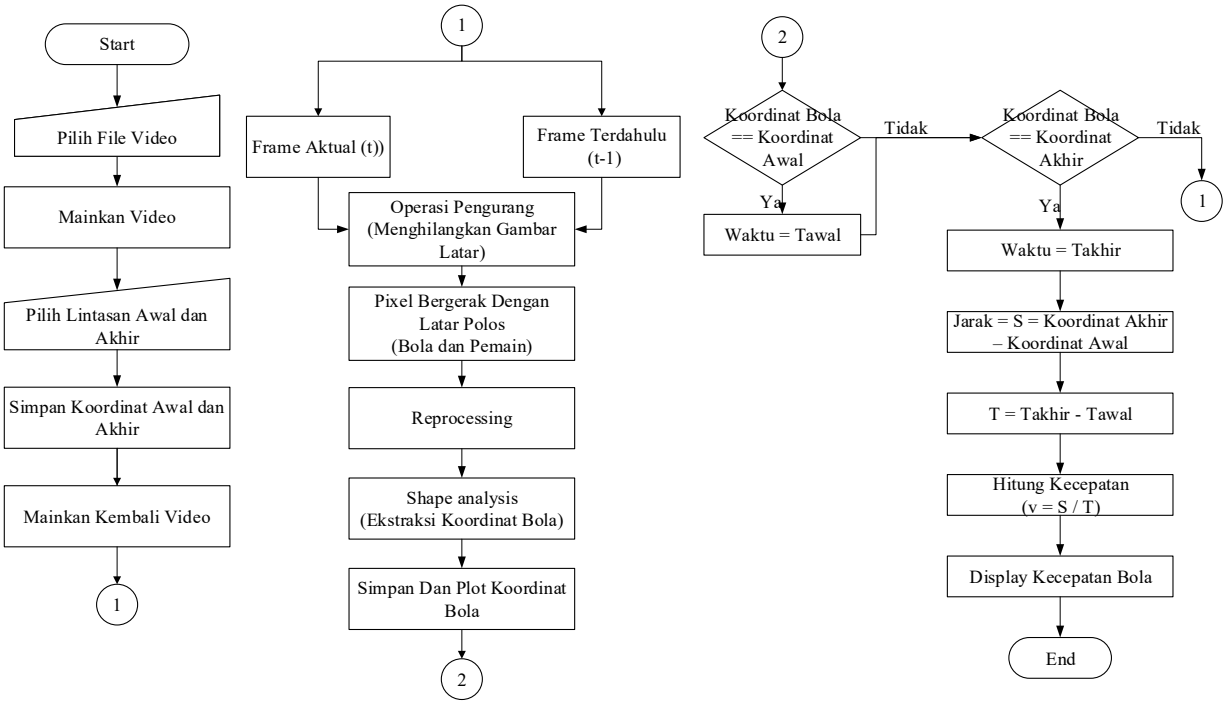
Jumlah *frame* dapat dicari dengan cara mengekstrak setiap *frame* saat bola terdeteksi pada kedua lintasan, lalu dihitung jumlah *frame* diantara kedua *frame* yang terdeteksi tersebut. Untuk mengetahui bola tedeteksi pada kedua lintasan maka akan dibandingkan koordinat *pixel contour self.kbx* dan *self.kby* ketika bola terdeteksi dalam daerah lintasan ke-1 x1, y1, x2 dan y2. Saat nilai *self.kbx* dan *self.kby* termasuk dalam *range* x1, x2, y1 dan y2 diambil urutan *frame* dalam kondisi tersebut dan disimpan dalam variabel *self.bingkai1*. Sama halnya dengan lintasan 1, pada lintasan 2 dibandingkan pula koordinat *pixel contour self.kbx* dan *self.kby* apakah termasuk dalam daerah lintasan 2 yaitu xr1, xr2, yr1 dan yr2. Jika ya maka selanjutnya diambil urutan *frame* dalam kondisi tersebut dan disimpan pada variabel *self.bingkai2*. Selanjutnya urutan *frame* yang telah didapat dihitung untuk mengetahui nilai Jmf dengan mengurangi nilai *self.bingkai2* dengan *self.bingkai1*.

2.7 Algoritma Menghitung Kecepatan

Kecepatan dihitung berdasarkan rumus kecepatan dalam km/jam yang telah didapatkan pada pembahasan sebelumnya. Agar program dapat menghitung kecepatan dengan rumus tersebut maka perlu dimasukan koordinat bola awal dan akhir, koordinat meja awal dan akhir, jumlah *frame* antara koordinat bola awal dan akhir, dan kecepatan video dalam fps. Hasil pengukuran kecepatan ditampilkan pada label kecepatan yaitu *self.textkec*, serta jarak dan waktu yang terukur ditampilkan pula pada label jarak dan waktu *self.textjarak* dan *self.textwaktu* agar dapat dilihat pada aplikasi.

2.8 Diagram Alir Program Aplikasi

Gambar 8 merupakan diagram alir keseluruhan program aplikasi ini. Alur proses diagram tersebut dimulai dari memilih *file* video pertandingan tenis meja yang terdapat pada komputer untuk dimasukan ke dalam program.



Gambar 8. Diagram alir keseluruhan program aplikasi

Selanjutnya ditentukan titik koordinat awal dan titik koordinat akhir bola sebagai masukan jarak yang ditempuh bola. Data koordinat tersebut disimpan dan kemudian video dimainkan kembali. Pada tahap ini video yang dimainkan dibandingkan setiap *frame* pada video tersebut yaitu *frame* aktual (t) dan *frame* terdahulu (t1). Untuk proses pendeteksian bola, kedua *frame* tersebut dikurangkan untuk menghilangkan gambar latar belakang yang terlihat pada video. Ketika gambar latar dihilangkan yang tersisa hanyalah pemain dan bola pada *frame* video tersebut. Pemain dan bola merupakan *pixel* yang

bergerak pada latar belakang polos yang kemudian dilakukan *preprocessing* dari hasil pergerakan *pixel* tersebut. Hasil yang didapatkan dilanjutkan dengan proses *shape* analisis untuk diekstrak koordinat bolanya sehingga menghasilkan plot lintasan bola. Koordinat dan lintasan yang didapat disimpan lalu dibandingkan apakah koordinat bola tersebut merupakan koordinat awal atau koordinat akhir. Jika koordinat bola sama dengan koordinat awal maka waktu yang didapat dimasukkan pada T_awal, sedangkan jika koordinat bola sama dengan koordinat akhir maka waktu dimasukkan sebagai T_akhir. Selisih koordinat awal dan akhir dijadikan sebagai jarak tempuh bola dan selisih waktu awal dan akhir dijadikan sebagai waktu bola melaju. Jarak dan waktu yang didapat dihitung menggunakan persamaan kecepatan dan ditampilkan pada halaman aplikasi lalu program berakhir.

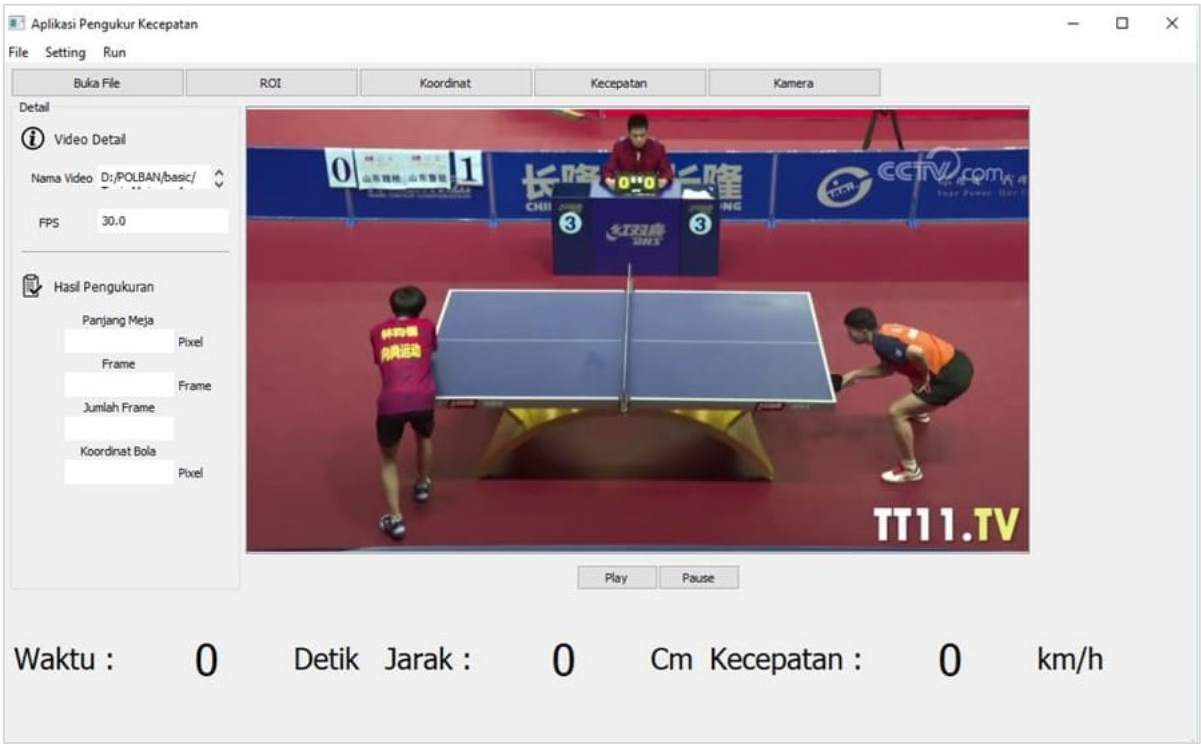
3. HASIL DAN PEMBAHASAN

3.1 Pengujian *User Interface* Aplikasi

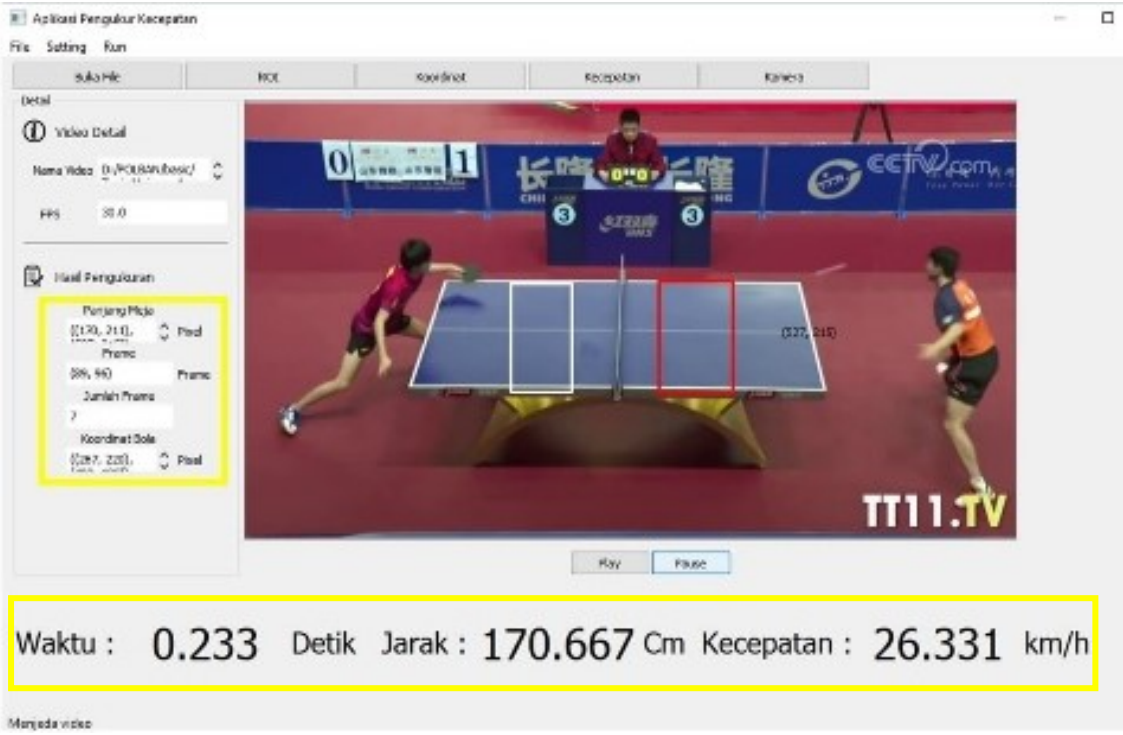
Dalam pengujian *user interface* aplikasi, program yang telah dibuat dijalankan dengan menekan tombol *run.* Pada pengujian ini pengguna harus menekan tombol buka *file* pada halaman aplikasi agar dapat memasukan video. Kemudian muncul *file dialog* untuk memilih *file* video yang ingin diukur kecepatan. Video yang dimasukan harus berformat MP4, *frame rate* 30 fps, sudut pandang video tampak atas atau tampak samping, dan terlihat dari sisi panjang meja dan lintasan bola. Lalu *file* video yang dipilih ditampilkan pada halaman aplikasi seperti pada Gambar 9.

Kemudian untuk memasukan koordinat meja dilakukan dengan menekan tombol koordinat. Selanjutnya menekan bagian meja permainan pada video yang dimasukan tersebut. Bagian meja yang ditekan akan diimunculkan koordinatnya.

Untuk menampilkan koordinat bola, meja, dan *frame* dilakukan dengan menekan tombol kecepatan kemudian tombol "*play*" untuk proses pengolahan video. Ketika bola terdeteksi pada kedua lintasan yang telah dibuat maka koordinat bola, meja, dan *frame* ditampilkan pada bagian hasil pengukuran seperti Gambar 10.



Gambar 9. Tampilan pengujian *user interface* aplikasi



Gambar 10. Tampilan pengujian untuk menampilkan data koordinat, waktu, jarak, dan kecepatan

3.2 Pengujian Program Penghitung Kecepatan

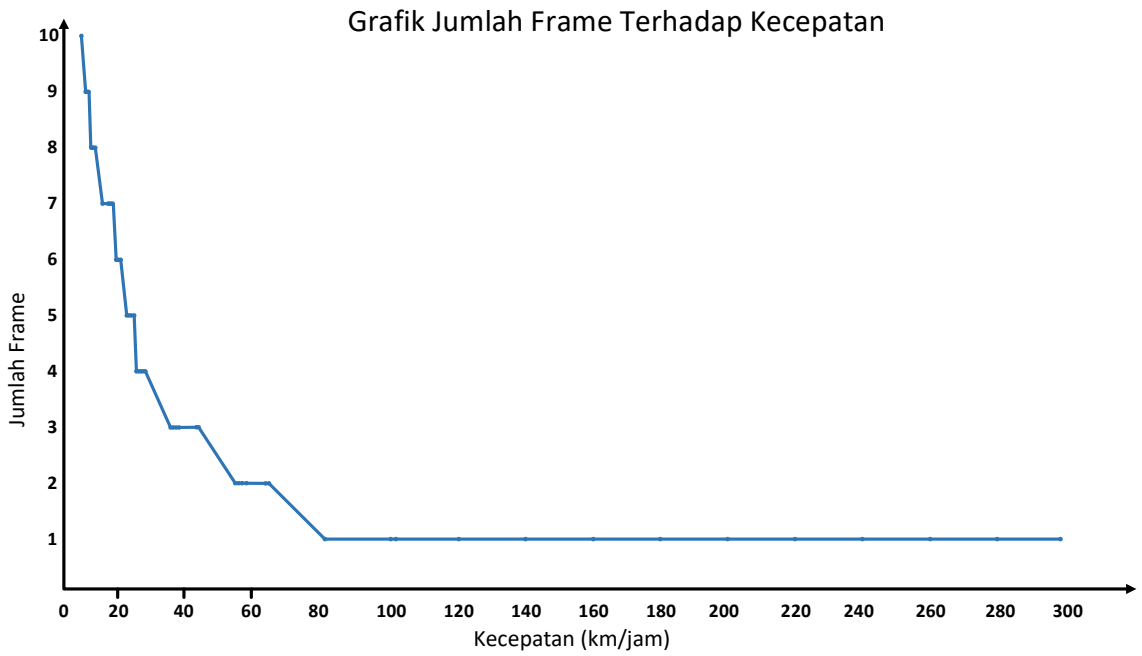
Dalam pengujian ini digunakan video *input* dengan latar belakang sama, namun diambil *sequence-sequence* video yang menunjukkan bola melintas dengan cepat. Selain itu digunakan juga video *dummy* untuk mendapatkan kecepatan yang lebih beragam. Video *dummy* adalah video normal yang di-*edit* agar jarak bola dan jumlah *frame* antara bola dapat dikondisikan untuk dapat menghasilkan kecepatan yang diinginkan ketika dimasukan pada program. Metode yang digunakan pada pengujian ini adalah metode hitung jumlah *frame* untuk mendapatkan waktu bola dan persamaan untuk mendapatkan jarak bola dilayar. Data hasil pengujian kecepatan bola ditampilkan pada Tabel 1.

Untuk kecepatan tertinggi yang didapat melalui video normal adalah 112,062 km/jam. Dikarenakan pada percobaan, bola dengan kecepatan-kecepatan tinggi jarang ditemukan, dan kalaupun ada, bola hanya terlihat di dua *frame* dan terputus sehingga agar tabel dapat dilengkapi maka beberapa *video dummy* dibuat. Terdapat 12 *video dummy* yang dibuat dari total 51 video yang digunakan untuk mengisi Tabel 1 agar grafik jumlah *frame* terhadap kecepatan dapat diselesaikan.

Gambar 11 adalah grafik hubungan antara jumlah *frame* dan kecepatan. Seperti terlihat pada gambar bahwa grafik tidak linier. Pada kecepatan antara 80 hingga 295 km/jam jumlah *frame* yang didapat sama dengan 1. Mengapa kecepatan bisa bervariasi, ini dikarenakan jarak pixel berbeda (lihat Tabel 1).

Tabel 1. Data kecepatan

No.	Nama *file*	Jmf	Vr	Jm_cm	Tb_dtk	Jb_cm	V (km/jam)	Keterangan
1	Tenis Meja	6	30	12,501	0,200	34,8	6,280	*Sequence video*
2	Tenis Meja.11	10	30	12,051	0,333	111,1	12,001	*Sequence video*
3	Tenis Meja.27	9	30	12,051	0,300	103,1	12,367	*Sequence video*
4	Tenis Meja.24	9	30	12,051	0,300	108,9	13,067	*Sequence video*
5	Tenis Meja.21	9	30	12,051	0,300	114,8	13,777	*Sequence video*
6	Tenis Meja.30	8	30	12,051	0,267	104,3	14,075	*Sequence video*
7	Tenis Meja.26	8	30	12,051	0,267	106,0	14,312	*Sequence video*
8	Tenis Meja.35	8	30	12,051	0,267	106,8	14,420	*Sequence video*
9	Tenis Meja.9	8	30	12,051	0,267	106,8	14,424	*Sequence video*
10	Tenis Meja.32	7	30	12,051	0,233	95,9	14,798	*Sequence video*
11	Tenis Meja.16	7	30	12,051	0,233	100,1	15,445	*Sequence video*
12	Tenis Meja	7	30	12,051	0,233	101,6	15,672	*Sequence video*
13	Tenis Meja.36	7	30	12,051	0,233	107,0	16,516	*Sequence video*
14	Tenis Meja.23	7	30	12,051	0,233	108,7	16,765	*Sequence video*
15	Tenis Meja.18	6	30	12,051	0,200	103,4	18,614	*Sequence video*
16	Tenis Meja.40	6	30	12,051	0,200	106,2	19,108	*Sequence video*
17	Tenis Meja.2	6	30	12,051	0,200	113,3	20,395	*Sequence video*
18	Tenis Meja.39	5	30	12,051	0,167	102,0	22,037	*Sequence video*
19	Tenis Meja.8	5	30	12,051	0,167	113,9	24,608	*Sequence video*
20	Tenis Meja.5	5	30	12,051	0,167	116,8	25,237	*Sequence video*
21	Tenis Meja.22	5	30	12,051	0,167	126,0	27,225	*Sequence video*
22	Tenis Meja.6	4	30	12,051	0,133	107,5	29,035	*Sequence video*
23	Tenis Meja.33	4	30	12,051	0,133	109,5	29,557	*Sequence video*
24	Tenis Meja.14	4	30	12,051	0,133	112,1	30,271	*Sequence video*
25	Tenis Meja.25	4	30	12,051	0,133	115,3	31,119	*Sequence video*
26	Tenis Meja.4	4	30	12,051	0,133	117,4	31,694	*Sequence video*
27	Tenis Meja.28	3	30	12,051	0,100	99,6	35,869	*Sequence video*
28	Tenis Meja.17	3	30	12,051	0,100	101,1	36,409	*Sequence video*
29	Tenis Meja.19	3	30	12,051	0,100	101,2	36,421	*Sequence video*
30	Tenis Meja.20	3	30	12,051	0,100	101,8	36,633	*Sequence video*
31	Tenis Meja.37	3	30	12,051	0,100	116,9	42,087	*Sequence video*
32	Tenis Meja.13	3	30	12,051	0,100	117,2	42,179	*Sequence video*
33	Tenis Meja.29	2	30	12,051	0,067	99,8	53,881	*Sequence video*
34	Tenis Meja.34	2	30	12,051	0,067	106,3	57,403	*Sequence video*
35	Tenis Meja.15	2	30	12,051	0,067	108,0	58,332	*Sequence video*
36	Tenis Meja.12	2	30	12,051	0,067	108,8	58,730	*Sequence video*
37	Tenis Meja.10	2	30	12,051	0,067	115,2	62,201	*Sequence video*
38	Tenis Meja.3	2	30	12,051	0,067	118,0	63,733	*Sequence video*
39	TM1*frame*2	1	30	12,051	0,033	74,3	80,248	*Video dummy*
40	TM1*frame*.47	1	30	12,051	0,033	93,4	100,868	*Video dummy*
41	Tenis Meja.1	1	30	12,051	0,033	103,8	112,063	*Sequence video*
42	TM1*frame*.45	1	30	12,051	0,033	112,0	120,930	*Video dummy*
43	TM1*frame*.42	1	30	12,051	0,033	130,5	140,992	*Video dummy*
44	TM1*frame*.31	1	30	12,051	0,033	148,6	160,497	*Video dummy*
45	TM1*frame*.28	1	30	12,051	0,033	166,7	180,001	*Video dummy*
46	TM1*frame*1	1	30	12,051	0,033	185,2	200,064	*Video dummy*
47	TM1*frame*.15	1	30	12,051	0,033	203,8	220,126	*Video dummy*
48	TM1*frame*.12	1	30	12,051	0,033	222,9	240,745	*Video dummy*
49	TM1*frame*.4	1	30	12,051	0,033	241,5	260,807	*Video dummy*
50	TM1*frame*.11	1	30	12,051	0,033	259,5	280,312	*Video dummy*
51	TM1*frame*.8	1	30	12,051	0,033	273,5	295,359	*Video dummy*



Gambar 11. Grafik jumlah *frame* terhadap kecepatan

3.3 Analisis dan Pembahasan

Walaupun sudah didapatkan kecepatan yang bervariasi dari 6 km/jam sampai dengan 295 km/jam, namun nilai kecepatan yang dihasilkan ini belum terkalibrasi dan belum sesuai dengan standar alat ukur kecepatan profesional seperti pada produk komersial [1], [2]. Membandingkan hasil pengukuran ini dengan alat ukur kecepatan bola profesional akan sulit dilakukan karena situasi pengukurannya berbeda dimana dalam penelitian ini menggunakan rekaman video (*offsite*), sedangkan pada pengukur kecepatan profesional dilakukan langsung di lapangan (*onsite*).

Teknik mencari nilai waktu pada aplikasi ini menggunakan teknik menghitung jumlah *frame* yang dikonversi ke satuan waktu fungsi dari kecepatan video dalam *frame* per detik (fps) yang digunakan. Hal ini cukup menarik dikarenakan waktu yang dihasilkan dari teknik ini tidak dipengaruhi oleh algoritma pengolahan citra yang digunakan dimana pada umumnya akan memperlambat proses penampilan video ataupun kalau ingin pada ritme penampilan normal harus mengurangi beberapa *frame* video. Lalu dikarenakan standar kecepatan video sudah terkalibrasi secara waktu maka perhitungan waktu yang dihasilkan pun akan tepat.

Kelemahan dari aplikasi ini terletak pada pengukuran jaraknya yang sangat dipengaruhi oleh arah bola bergerak dalam bidang 3 dimensi dimana pada kenyatannya perekaman video dibuat dalam format 2 dimensi sehingga pergerakan bola pada sumbu Z tidak dapat diketahui. Bila pergerakan bola tidak melibatkan sumbu Z (Z = 0) dan searah dengan panjang benda yang dijadikan acuan (panjang meja), maka kecepatan yang dihasilkan cukup representatif. Untuk meminimalisir kesalahan jarak akibat sumbu Z, maka video yang diolah dipilih pada sudut pandang samping seperti yang telah dilakukan. Tetapi sebagai konsekuensi dari itu, kecepatan bola yang tinggi sulit ditangkap *oleh frame-frame* video ini ditandai dengan adanya lintasan bola terputus putus bahkan bola sempat menghilang dan hanya tampak di 2 titik saja. Berbeda dengan jika sudut pandang tampak belakang atas yang mana bola akan selalu tampak pada layar dan tidak pernah menghilang. Konsekuensi dari itu adalah ada beberapa video dengan kecepatan tertentu sulit didapatkan sehingga sulit untuk melengkapi Tabel 1. Untuk itu maka solusinya adalah dengan membuat semacam *dummy* video (video tiruan) dimana kita dapat menyisipkan keberadaan bola pada posisi-posisi tertentu sehingga ketika video ini diproses akan menghasilkan kecepatan yang dinginkan sehingga dapat mengisi kekosongan pada Tabel 1. Nilai kecepatan yang dihasilkan menunjukkan perbedaan yang jelas dan tegas antara bola lambat, sedang dan cepat pada video yang sama atau pada video yang berbeda. Nilai ini dapat bervariasi antara 6 km/jam hingga 295 km/jam. Dikarenakan aplikasi ini hanya dipergunakan sebagai sarana membandingkan seberapa cepat bola satu dengan bola-bola yang lainnya dalam satu video yang sama atau dalam video pertandingan tenis meja yang berbeda namun yang menggunakan format kecepatan

video dan sudut pandang yang sama, maka perbedaan nilai kecepatan yang dihasilkan terhadap pengukur kecepatan profesional tidak menjadi kendala bagi pengguna.

4. KESIMPULAN

Aplikasi yang dibuat telah berhasil mengukur dan mengkuantisasi nilai waktu, jarak, dan kecepatan bola dari sebuah video permainan tenis meja. Nilai kecepatan yang dihasilkan bervariasi antara 6 km/jam hingga 295 km/jam. Nilai kecepatan yang dihasilkan ini belum dapat dikatakan akurat karena belum dilakukan perbandingan dengan alat ukur kecepatan standar profesional. Namun nilai kecepatan yang didapat menunjukkan perbedaan yang jelas antara bola lambat, sedang, dan cepat dengan angka-angka. Untuk penelitian selanjutnya, aplikasi akan lebih baik bila menggunakan video yang diambil dengan kamera berkecepatan tinggi (di atas 30 fps). Untuk menghindari lintasan bola yang kosong, maka video yang digunakan sebaiknya dengan sudut pandang tampak belakang atas dikarenakan bola selalu tampak dilayar.

REFERENSI

[1] G. Robinson and I. Robinson, "Radar speed gun true velocity measurements of sports-balls in flight: application to tennis*," Physica Scripta*, vol. 91, no. 2, pp. 023008, 2016.

[2] R. Thakur, M. Qureshi, S. Sarile, S. Pandit, and L. Tahilani, "Smart Vehicle Monitoring and Control System using Arduino and Speed Gun: A Case Study," *Computing Technologies and Applications*, Chapman and Hall/CRC, pp. 279-300, 2021.

[3] R. M. Antúnez, F. J. M. Hernández, J. P. F. García, R. R. Vaíllo, and J. S. D. Arroyo, "Relationship Between Motor Variability, Accuracy, and Ball Speed In The Tennis Serve," *Journal of Human Kinetics*, vol. 33, pp. 45-53, 2012.

[4] F. Ramdani, "Penerapan Object Tracking Dengan Metode Adaptive Particle Filter Untuk Pelacakan Bola Pada Permainan Tenis," *Tugas Akhir Universitas Pendidikan Indonesia*, 2017.

[5] A. Fahriannur, R. Mardiyanto, and M. Siswanto, "Sistem Pelacakan Objek Menggunakan Kombinasi Algoritma Optical Flow dan Template Matching," *Jurnal Teknologi dan Sistem Komputer*, vol. 6, no. 1, pp. 13-17, 2018.

[6] N. Khamdi, M. Susantok, and P. Leopard, "Pendeteksian Objek Bola dengan Metode Color Filtering HSV pada Robot Soccer Humanoid," *Jurnal Nasional Teknik Elektro*, vol. 6, no. 2, pp. 123-128, 2017.

[7] M. S. Sabirin, R. Magdalena, and E. Susatio, "Prototipe Pendeteksi Jarak Menggunakan Kamera Smartphone Dan Laser Dengan Model Warna HSV," *eProceedings of Engineering*, vol. 7, no. 2, 2020.

[8] A. M. Rizal, M. Taufiqurrohman, and J. Subur, "Implementasi SIstem Tracking Bola Pada Pengolahan Citra Menggunakan Metode Find Contour," *Jurnal of Electrical and Electronic Engineering-UMSIDa*, vol. 4, no. 2, pp. 136-149, 2020.

[9] D. E. W. Putra, "Deteksi dan Tracking Objek Berbentuk Bola Menggunakan Kalman Filter," *Reporsitory Universitas Dinamika*, 2017.

[10] E. M. Pamungkas, B. A. A. Sumbodo, and I. Candradewi, "Sistem Pendeteksi dan Pelacakan Bola dengan Metode Hough Circle Transform, Blob Detection, dan Camshift Menggunakan AR.Drone," *IJEIS*, vol. 7, no. 1, pp. 1-12, 2017.