

# LAPORAN TUGAS INDIVIDU 1

## IF5152 COMPUTER VISION

Aplikasi Sederhana Integratif:  
Materi Minggu 3-6


Oleh:

23525041

Versa Syahputra Santo

Sekolah Teknik Elektro dan Informatika - Institut  
Teknologi Bandung

Jl. Ganesha 10, Bandung 40132

	Sekolah Teknik Elektro dan Informatika ITB	Nama Dokumen
		IF5152_CompVis-K01_23525041.pdf

## Daftar Isi

1	Workflow Pipeline	3
2	Proses dan Hasil tiap Fitur	4
2.1	<i>Filtering</i>	4
2.2	<i>Edge Detection</i>	7
2.3	<i>Feature Points</i>	9
2.4	<i>Camera Geometry</i>	13
3	Komparasi dan Refleksi Pribadi	15
3.1	Komparasi Umum Hasil	15
3.2	Refleksi Pribadi	16
4	Lampiran	17
4.1	Link folder proyek	17

# 1 Workflow Pipeline

Aplikasi ini memiliki empat tahap utama: penyaringan, deteksi tepi, deteksi titik fitur, dan geometri/kalibrasi. Keempat tahap ini dirancang sebagai modul notebook yang dapat dijalankan secara terpisah. Namun secara konseptual, modul-modul ini membentuk alur kerja yang mengikuti urutan tertentu, mirip dengan sistem penglihatan komputer yang sebenarnya.

## **Tahap 1 - Filtering**

Langkah ini mengurangi noise menggunakan filter Gaussian. Selain itu, juga dieksplorasi filter lain, seperti filter median yang bersifat non-linear. Hal ini membuat gambar menjadi lebih halus. Langkah ini juga digunakan untuk mempersiapkan gambar untuk tahap berikutnya, yaitu deteksi tepi.

## **Tahap 2: Deteksi Edge**

Tahap ini menggunakan dua metode, Sobel dan Canny, untuk mendeteksi tepi objek. Saat mengimplementasikan proses ini, gambar yang digunakan dapat berasal dari hasil penyaringan (untuk mengurangi noise) atau gambar asli, tergantung pada eksperimen parameter. Namun, metode detektor tepi Canny akan secara otomatis menggunakan filter Gaussian Blur sebelum melakukan perhitungan gradien.

## **Tahap 3: Deteksi Feature Points**

Langkah ini menggunakan Harris Corner dan FAST untuk menandai titik-titik penting. Tahap ini dapat dijalankan dengan dua cara: dari gambar asli atau dari gambar peta tepi sebelumnya untuk eksperimen lebih lanjut. Namun, detektor tepi Harris juga menggunakan operator Sobel untuk mendeteksi tepi sebelum melakukan deteksi sudut.

## **Tahap 4 - Geometri/Kalibrasi**

Menggunakan pola papan catur untuk memperkirakan transformasi geometris. Outputnya mencakup overlay transformasi dan matriks parameter perkiraan.

Desain modular ini memungkinkan setiap konsep dievaluasi secara terpisah, sementara proses keseluruhan mengikuti urutan alur kerja logis yang bertahap.

## 2 Proses dan Hasil tiap Fitur

### 2.1 *Filtering*

Proses filtering citra bertujuan untuk meningkatkan kualitas citra dengan mengubah kecerahan setiap piksel berdasarkan piksel di sekitarnya. Proses ini dapat mengurangi noise pada citra, menonjolkan detail penting, dan membuat citra menjadi lebih halus dan stabil untuk analisis lebih lanjut. Filtering juga merupakan langkah penting sebelum tahap lain dalam proses pengolahan citra komputer. Hasil dari langkah ini memengaruhi seberapa baik tahap lain dalam proses tersebut berfungsi. Misalnya, hasilnya memengaruhi deteksi tepi, ekstraksi fitur, dan kalibrasi geometri.

Dua metode filtering yang diuji pada project ini adalah Gaussian Filter dan Median Filter.

#### 1. Gaussian Filter

Gaussian filter merupakan filter linear yang menggunakan fungsi distribusi Gaussian sebagai kernel. Tujuannya adalah menghaluskan citra dan mengurangi noise frekuensi tinggi.

Parameter utama:

- sigma ( $\sigma$ ): standar deviasi distribusi Gaussian, berfungsi untuk mengontrol tingkat blur.
- kernel\_size: ukuran jendela filter.

#### 2. Median Filter

Median filter adalah filter non-linear yang mengganti setiap piksel dengan nilai median dari area sekitarnya. Cocok untuk menghilangkan salt-and-pepper noise tanpa terlalu mengaburkan tepi citra.

Parameter utama:

- disk\_size: radius area yang digunakan untuk menghitung median.

Data Citra yang digunakan pada tahap ini adalah cameraman dan astronot dari skimage.data.

#### Original Images

Grayscale Image



Color Image



#### Tabel parameter uji untuk Gaussian Filter

Parameter	Values		
Sigma	0.5	1	2
Kernel Size	3	5	7

#### Tabel parameter uji untuk Median Filter

Parameter	Values		
Disk	1	3	5

Hasil terbaik Gaussian Filter ketika  $\sigma = 0.5$  dan kernel size = 3.

Gaussian Filtered Images ( $\sigma=0.5$ , kernel=3)

Grayscale Image



Color Image



Hasil terbaik Median Filter ketika disk size = 1

Median Filtered Images (disk=1)

Grayscale Image



Color Image



Secara umum, kedua metode menunjukkan bahwa:

- Parameter yang terlalu besar menyebabkan gambar menjadi terlalu buram, karena informasi detail ikut tersaring.
- Parameter yang tepat ( $\sigma=0.5$ , kernel=3, disk=1) menghasilkan gambar yang lebih jernih dan bersih, dengan noise berkurang namun detail masih terjaga.

Dengan demikian, baik Gaussian maupun Median filter sama-sama efektif untuk proses denoising, tetapi Median Filter lebih unggul pada noise acak ekstrem, sedangkan Gaussian Filter lebih cocok untuk smoothing halus secara umum.

## 2.2 Edge Detection

Deteksi tepi bertujuan menemukan perubahan intensitas tajam pada citra yang menandakan batas antar objek. Tepi merepresentasikan perubahan gradien (derivatif/turunan) intensitas piksel.

Dua metode Edge Detection yang diuji pada project ini adalah Sobel Operator dan Canny Edge Detection.

### 1. Sobel Operator

Metode Sobel menggunakan dua kernel konvolusi ( $G_x$  dan  $G_y$ ) untuk menghitung gradien arah horizontal dan vertikal.

Parameter utama: **kernel\_size**, ukuran kernel (3, 5, atau 7). Kernel besar menghasilkan tepi lebih halus tapi bisa kehilangan detail.

### 2. Canny Edge Detector

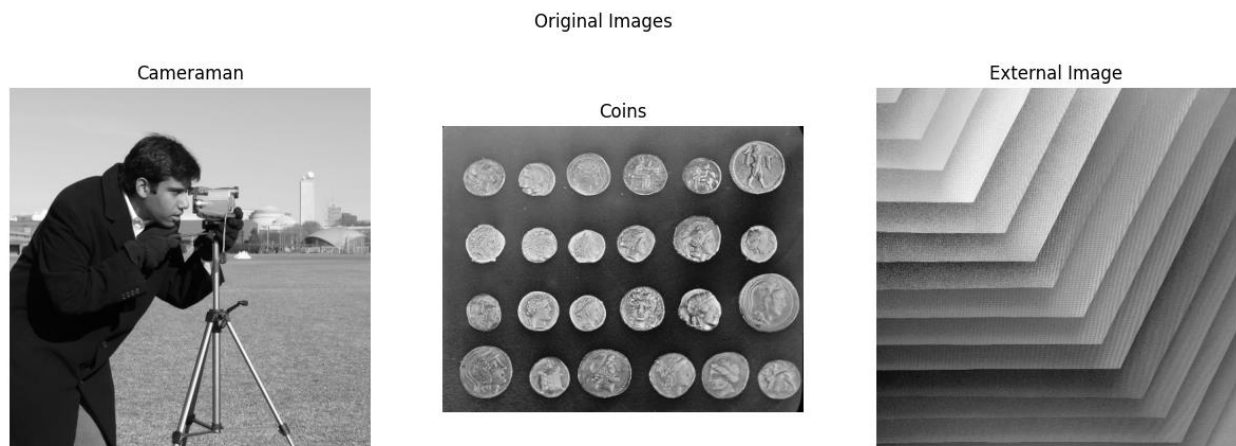
Metode Canny lebih kompleks, terdiri dari empat tahap utama:

1. Gaussian blur → mengurangi noise pada citra
2. Perhitungan gradien intensitas → mendeteksi perubahan arah tepi (mirip Sobel)
3. Non-maximum suppression → mempertahankan hanya tepi yang paling tajam
4. Hysteresis thresholding → menentukan tepi kuat dan lemah berdasarkan dua nilai ambang

Parameter utama:

- threshold1 (low) : batas bawah intensitas gradien.
- threshold2 (high): batas atas intensitas gradien.
- apertureSize (opsional): ukuran kernel Sobel internal (default = 3).

Data Citra yang digunakan pada tahap ini adalah cameraman dan coins dari skimage.data dengan tambahan 1 gambar external oleh [Clark Van Der Beken from Pexels](#). Gambar external tersebut diubah menjadi *grayscale* dan direduksi ukurannya menjadi 512x512 piksel.



Tabel parameter uji untuk Sobel Edge

Parameter	Values		
Kernel Size	3	5	7

Tabel parameter uji untuk Canny Edge

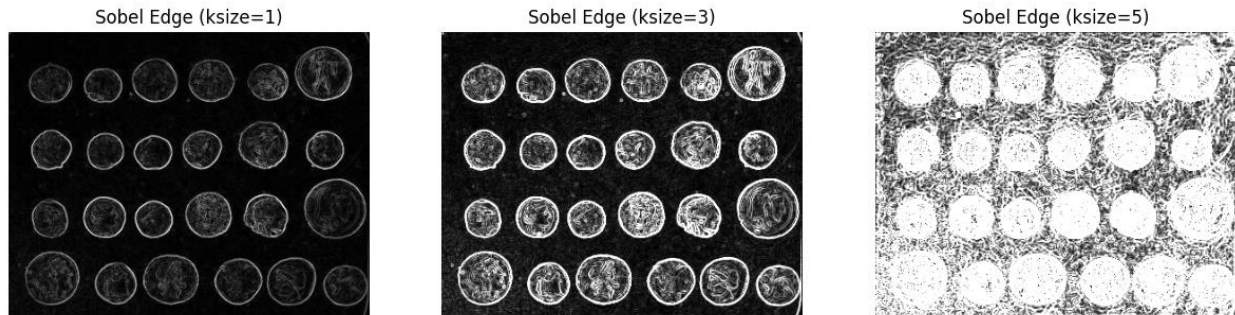
Parameter	Values		
High Threshold	100	200	250
Low Threshold	50	100	150

Gambar hasil Sobel Operator berikut menunjukkan bahwa ukuran kernel paling seimbang adalah 3. Jika ukuran kernel terlalu besar



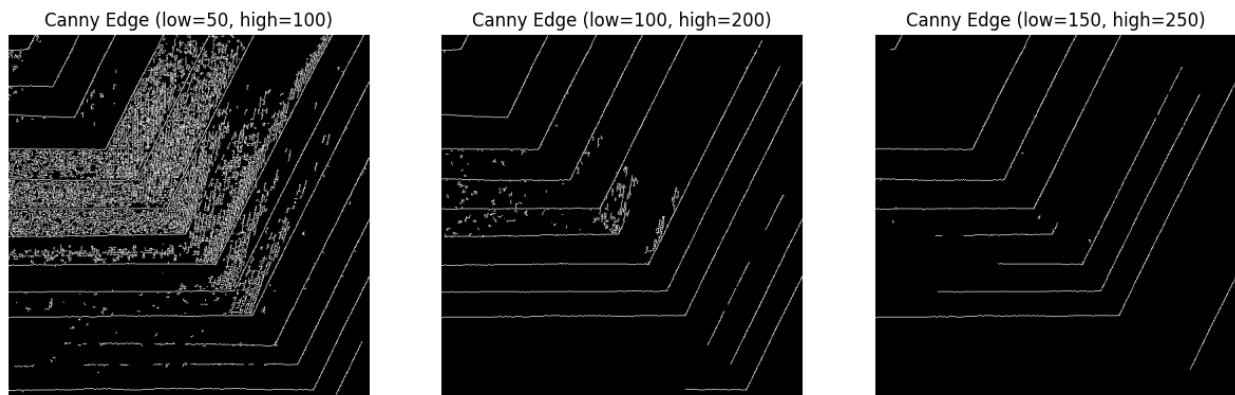
maka semakin banyak noise yang terdeteksi sebagai edge. Namun apabila terlalu kecil maka beberapa tepi halus tidak terdeteksi.

Sobel Edge Detection - Coins



Hal yang sama juga berlaku pada Canny edge, namu parameter yang memengaruhi adalah threshold. Jika *low threshold* terlalu rendah maka akan banyak noise yang tertangkap sebagai edge. Sebaliknya jika *high threshold* terlalu tinggi maka hanya tepi dengan kontras yang kuat yang terdeteksi sehingga banyak tepi halus yang terabaikan.

Canny Edge Detection - External



## 2.3 Feature Points

Deteksi titik fitur (feature points) bertujuan menemukan bagian-bagian citra yang memiliki **pola lokal unik** dan **stabil terhadap perubahan** seperti rotasi, skala, atau pencahayaan. Titik-titik ini biasanya terletak pada **sudut**, **ujung pola**, atau **area dengan variasi intensitas tinggi**, sehingga dapat menjadi penanda (strong interest points) untuk mengenali atau mencocokkan citra.

Dua metode deteksi Feature Points yang diuji pada project ini adalah Harris Corner Detector dan FAST (Features from Accelerated Segment Test).

## 1. Harris Corner Detector

Metode ini bertujuan untuk mendeteksi **sudut (corner)** pada citra, yaitu titik di mana intensitas berubah signifikan ke dua arah ortogonal. Titik ini biasanya menandakan struktur penting seperti ujung objek, jendela, atau perpotongan tepi.

Parameter utama:

- blockSize: ukuran jendela lokal untuk menghitung gradien.
- ksize: ukuran kernel Sobel.
- k: konstanta sensitivitas (umumnya 0.04-0.06).

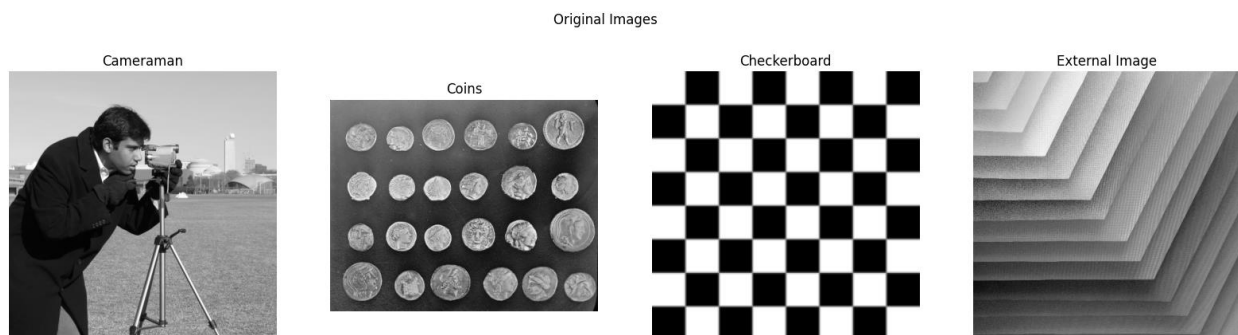
## 2. FAST (Features from Accelerated Segment Test)

Metode ini bertujuan mendeteksi *interest points* secara cepat dengan membandingkan intensitas piksel pusat terhadap piksel-piksel sekitarnya. FAST dioptimalkan untuk kecepatan dan cocok untuk aplikasi real-time, meskipun lebih sensitif terhadap noise.

Parameter yang memengaruhi:

- threshold: sensitivitas perbedaan intensitas.
- nonmaxSuppression: menekan fitur duplikat (boolean).

Empat jenis citra digunakan pada tahap ini, yaitu cameraman, coins, dan checkerboard dari skimage.data dengan tambahan 1 gambar external oleh [Clark Van Der Beken from Pexels](#).



### 3. Tabel parameter uji untuk Harris Corner Detector

Parameter	Values		
Block Size	2	2	4
Kernel Size	3	3	3
Konstanta Sensitivitas k	0.04	0.06	0.04

### 4. Tabel parameter uji untuk FAST Feature Detector

Parameter	Values		
Threshold	10	25	50
Non Max Suppression	True/False		

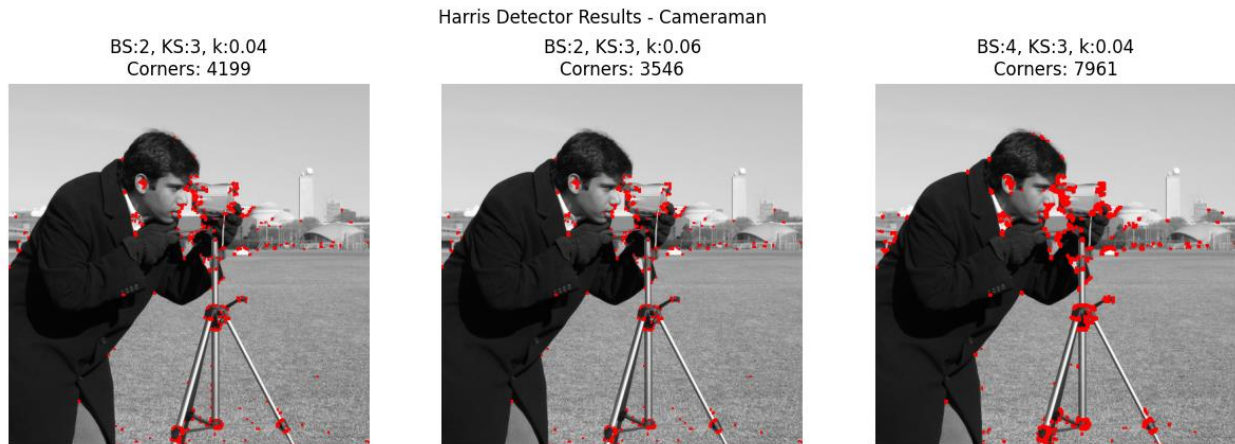
### 5. Analisis Metode Harris Corner Detector

Pada eksperimen dengan variasi parameter **block size** dan konstanta **k**, diperoleh bahwa:

- Semakin besar nilai `blockSize`, semakin banyak area lokal yang dianalisis, sehingga jumlah sudut (corner) yang terdeteksi semakin banyak. Hal ini terjadi karena area analisis yang lebih luas membuat perubahan intensitas di sekitar piksel lebih mudah terdeteksi sebagai sudut, meskipun sebagian di antaranya bisa berupa *false positive*.
- Peningkatan nilai konstanta **k** menyebabkan hasil deteksi menjadi lebih selektif, sehingga jumlah corner berkurang. Nilai **k** yang lebih tinggi membuat algoritma lebih ketat dalam menilai apakah suatu titik benar-benar memiliki perubahan intensitas yang signifikan ke dua arah ortogonal.
- Namun, pada citra checkerboard, perubahan nilai **k** hampir tidak berpengaruh signifikan terhadap jumlah corner. Hal ini disebabkan oleh pola kotak catur yang memiliki perbedaan intensitas yang sangat jelas dan teratur, sehingga sudut-sudutnya tetap kuat terdeteksi bahkan dengan parameter sensitivitas yang lebih tinggi.
- Menariknya, pada gambar eksternal, perubahan nilai **k** tidak menurunkan jumlah corner secara berarti seperti halnya pada citra camera dan coin. Kemungkinan hal ini terjadi karena distribusi tepi dan tekstur pada gambar eksternal relatif

tidak kompleks, sehingga perbedaan sensitivitas  $k$  tidak menghasilkan variasi respons yang besar.

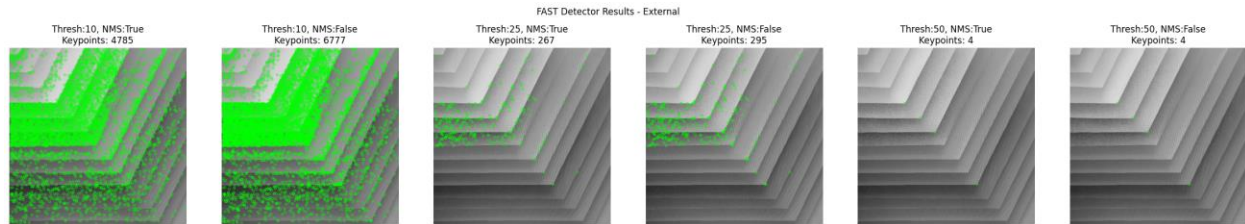
Secara keseluruhan, metode Harris bekerja baik untuk mendeteksi struktur geometris yang jelas seperti pada checkerboard dan camera, namun parameter perlu diatur hati-hati agar tidak terlalu banyak mendeteksi area kontras lemah sebagai corner.



## 6. Analisis Metode FAST Feature Detector

Eksperimen terhadap parameter **threshold** dan penggunaan **non-maximum suppression (NMS)** menunjukkan bahwa:

- Nilai threshold yang kecil membuat algoritma FAST menjadi sangat sensitif terhadap perbedaan intensitas, sehingga banyak keypoints terdeteksi, termasuk yang berasal dari noise.
- Sebaliknya, threshold yang terlalu besar membuat algoritma melewati banyak area potensial, bahkan dapat gagal mendeteksi fitur sama sekali. Hal ini terlihat jelas pada gambar eksternal, di mana hanya empat keypoint terdeteksi ketika threshold = 50, sementara lebih dari empat ribu titik terdeteksi pada threshold = 10.
- Penggunaan Non-Max Suppression (NMS) juga berpengaruh penting terhadap hasil deteksi. Tanpa NMS, beberapa keypoint yang berdekatan (pada area kontras tinggi) akan dianggap fitur terpisah, sehingga jumlah total titik meningkat karena terjadi duplikasi deteksi pada area yang sama.



Dengan demikian, metode FAST cenderung lebih cepat dan praktis dibanding Harris, tetapi lebih sensitif terhadap parameter threshold dan lebih mudah terpengaruh oleh noise jika tidak disertai NMS. Pemilihan parameter yang tepat menjadi kunci agar jumlah fitur tidak berlebihan sekaligus tetap representatif terhadap struktur citra.

## 2.4 Camera Geometry

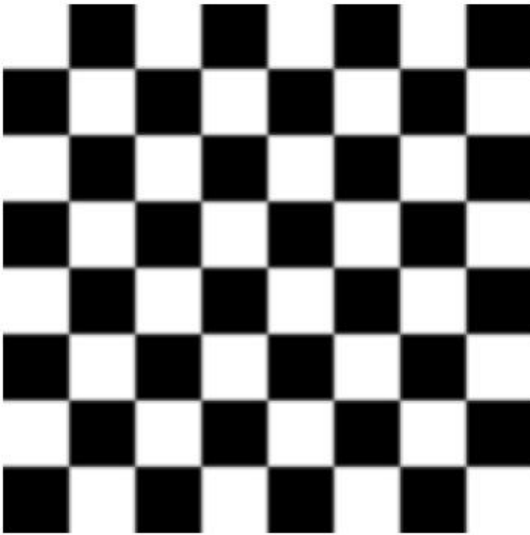
Camera Geometry & Calibration bertujuan untuk mengetahui dan mengoreksi bagaimana kamera “melihat” dunia agar kita bisa memahami posisi, bentuk, dan proyeksi objek di dalam gambar.

Geometry pada citra berhubungan dengan transformasi geometris pada citra, yaitu bagaimana koordinat piksel dapat diubah melalui operasi seperti translasi, rotasi, scaling, dan homografi.

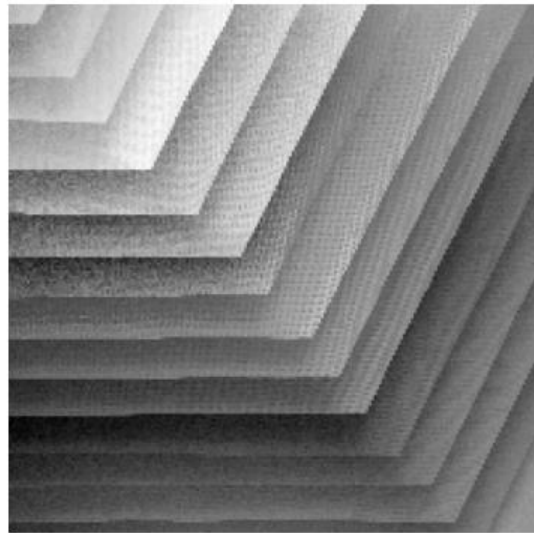
Pada tahap ini, 2 jenis citra digunakan yaitu checkerboard dari skimage.data dan gambar external oleh [Clark Van Der Beken from Pexels](#) yang direduksi ukurannya menjadi 200x200 agar sesuai.

## Sample Images

Checkerboard

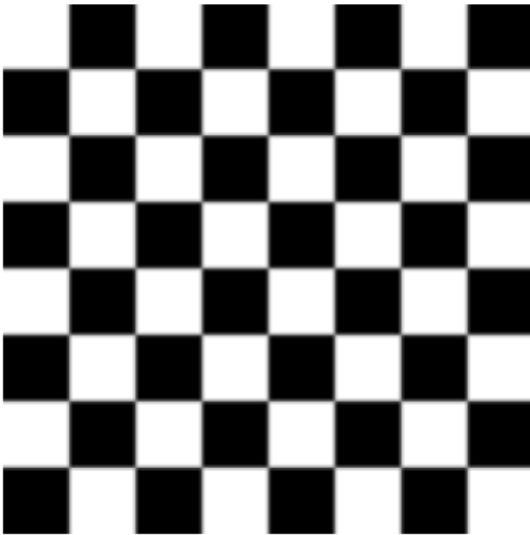


External Image

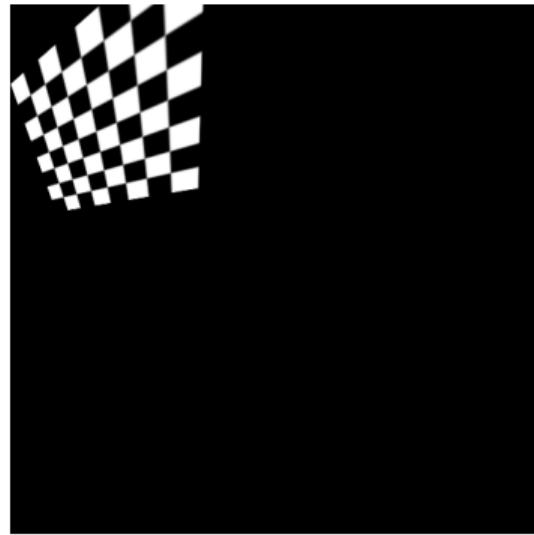


Hasil dari transformasi perspektif pada citra checkerboard:  
Homography Transformation

Original Checkerboard



Warped Checkerboard



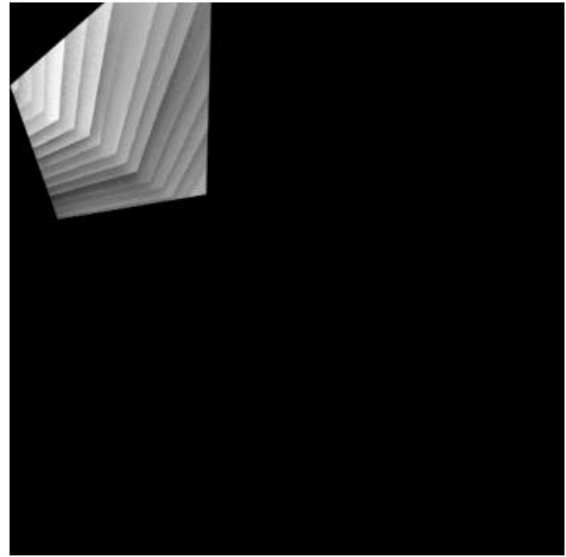
Hasil dari transformasi perspektif pada citra external:

## Homography Transformation for Ext image

Original image



Warped image



### Kesimpulan:

Pada bagian ini, diterapkan transformasi geometris menggunakan homografi pada citra checkerboard untuk mensimulasikan perubahan sudut pandang kamera. Hasil transformasi menunjukkan bahwa bidang gambar dapat diproyeksikan ke bentuk baru yang miring sesuai perubahan titik tujuan, merepresentasikan efek perspektif secara visual.

Overlay titik sumber dan tujuan memperlihatkan hubungan spasial antar bidang serta berfungsi sebagai validasi urutan titik korespondensi. Eksperimen ini membuktikan bahwa konsep geometri kamera memungkinkan pemetaan posisi piksel secara presisi, yang menjadi dasar bagi proses kalibrasi kamera dan aplikasi visi komputer lanjutan.

## 3 Komparasi dan Refleksi Pribadi

### 3.1 Komparasi Umum Hasil

Secara keseluruhan, seluruh metode menunjukkan bahwa citra standar dari skimage.data (seperti cameraman, coins, astronaut, dan checkerboard) memberikan hasil yang lebih stabil dan mudah dikontrol karena memiliki pencahayaan serta kontras yang ideal.

Sebaliknya, gambar eksternal memiliki variasi intensitas dan tekstur yang tinggi, sehingga memerlukan penyesuaian parameter yang lebih sensitif, terutama pada tahap edge detection dan feature points.

## 3.2 Refleksi Pribadi

Selama proses pengerjaan, saya menyadari bahwa setiap tahap dalam *Computer Vision pipeline* memiliki **ketergantungan kuat terhadap kualitas keluaran tahap sebelumnya**.

Eksperimen ini memperlihatkan pentingnya memilih parameter yang tepat untuk setiap jenis citra, serta perlunya pemahaman praktis tentang bagaimana filter dan detektor bekerja secara matematis maupun visual.

Beberapa refleksi dan keputusan pribadi:

- Saya memilih pendekatan **notebook terpisah per tahap** agar setiap bagian bisa diuji dan dianalisis secara independen, meskipun secara logis tetap membentuk satu alur pipeline terintegrasi.
- Saya menemukan bahwa **Canny** menggunakan opencv secara internal menggunakan filtering Gaussian. Hal ini menyebabkan pipeline aplikasi computer vision saling bergantung satu sama lain dari filtering -> Edge detection -> Feature points detection
- Pada tahap feature detection, saya memilih untuk mempertahankan **Harris dan FAST** karena memberikan keseimbangan antara presisi dan kecepatan. Selain itu, dua metode tersebut sudah *include* ke dalam modul opencv core sehingga tidak perlu menginstal modul extension opencv.
- Untuk *geometry*, saya belajar bahwa *homography* sangat bergantung pada pemilihan titik korespondensi yang akurat – sedikit kesalahan saja sudah memengaruhi hasil proyeksi secara signifikan.

Dari keseluruhan proses ini, saya memperoleh wawasan bahwa *Computer Vision pipeline* bukan hanya soal menerapkan algoritma, tetapi juga mengatur urutan, parameter, dan interpretasi hasil secara adaptif terhadap konteks citra.



## 4 Lampiran

### 4.1 Link folder projek

Google Drive:

<https://drive.google.com/drive/folders/1G0e11e0AJ0iEk5AFwLhf0ZGKjKIRn1Pr?usp=sharing>

Alternatif (Github):

<https://github.com/versa-syahptr/IF5152-Computer-Vision-Tugas-Individu-1>