# Proposal Teknis Pengembangan Sistem Persepsi RGB-D Amodal untuk Estimasi Fenotipe dan Penghitungan Buah Presisi Berbasis Modifikasi Arsitektur YOLOv8

## 1. Pendahuluan

Dalam era pertanian presisi (*precision agriculture*), kemampuan untuk memantau status tanaman, memperkirakan hasil panen (*yield estimation*), dan melakukan fenotyping otomatis menjadi sangat krusial. Salah satu tantangan teknis terbesar dalam visi komputer pertanian adalah deteksi dan pengukuran buah di lingkungan kebun yang tidak terstruktur, di mana oklusi (penghalangan pandangan) oleh daun, dahan, atau buah lain adalah norma, bukan pengecualian. Laporan ini mengajukan sebuah proposal teknis mendalam dan *low-level* untuk pengembangan sistem visi komputer hibrida yang mengintegrasikan data RGB (warna) dan Depth (kedalaman) menggunakan arsitektur *Deep Learning* yang dimodifikasi.

Sistem yang diusulkan bertujuan untuk mengatasi keterbatasan mendasar dari deteksi objek 2D konvensional. Pendekatan konvensional seringkali gagal dalam menghitung buah yang tertutup sebagian, tidak mampu memberikan ukuran fisik (dalam milimeter) tanpa referensi eksternal yang kompleks, dan rentan terhadap kesalahan penghitungan ganda (*double counting*) saat kamera bergerak. Solusi yang kami tawarkan adalah modifikasi radikal pada arsitektur YOLOv8 (*You Only Look Once, Version 8*) untuk menerima input 4-channel (RGB-D) dan menghasilkan segmentasi instan amodal (*amodal instance segmentation*), yang memprediksi bentuk utuh buah meskipun sebagian tertutup. Selanjutnya, output ini diproses melalui pipa pelacakan 3D (*3D tracking pipeline*) yang memanfaatkan geometri proyektif untuk memastikan akurasi penghitungan dan pengukuran dimensi fisik yang presisi.

Dokumen ini disusun sebagai panduan implementasi teknis yang komprehensif, mencakup landasan teoritis pemrosesan sinyal kedalaman, modifikasi tensor pada tingkat lapisan jaringan saraf, formulasi matematis untuk proyeksi 3D, hingga logika algoritma pelacakan.

## 2. Landasan Teori dan Analisis Masalah

Sebelum masuk ke detail implementasi, sangat penting untuk membedah tantangan fisik dan komputasi yang dihadapi dalam pemrosesan citra agrikultur, serta mengapa pendekatan standar tidak memadai.

### 2.1. Keterbatasan Segmentasi Modal vs. Amodal

Dalam visi komputer standar, tugas segmentasi instan biasanya menghasilkan "masker modal" (*modal mask*). Masker modal hanya mencakup piksel-piksel objek yang *terlihat* oleh kamera. Dalam konteks pertanian, ini menjadi masalah fatal. Jika sebuah apel tertutup 40% oleh daun, masker modal hanya akan memberikan bentuk sabit atau bentuk tidak beraturan yang merepresentasikan 60% sisanya.

Jika kita menggunakan masker modal ini untuk estimasi ukuran (misalnya, menghitung diameter mayor), hasilnya akan bias ke bawah secara signifikan. Kita tidak dapat membedakan antara buah kecil yang terlihat utuh dan buah besar yang tertutup sebagian.

Solusinya adalah "Segmentasi Amodal" (*Amodal Segmentation*). Persepsi amodal adalah kemampuan kognitif (yang juga dimiliki manusia) untuk memperkirakan bentuk keseluruhan objek berdasarkan bagian yang terlihat dan pengetahuan prior tentang bentuk objek tersebut.1 Dalam sistem yang diusulkan, jaringan saraf tidak hanya dilatih untuk mendeteksi piksel yang terlihat, tetapi juga untuk "berhalusinasi" atau merekonstruksi bagian yang hilang (oklusi) berdasarkan konteks visual yang ada, menghasilkan masker penuh yang merepresentasikan bentuk buah seolah-olah tidak ada penghalang.

### 2.2. Pentingnya Modalitas Kedalaman (Depth)

Citra RGB standar sangat rentan terhadap variasi pencahayaan. Di kebun, bayangan tajam dari sinar matahari atau kondisi cahaya rendah saat senja dapat mengaburkan tekstur buah, menyebabkan kegagalan deteksi.

Integrasi data kedalaman (Depth) memberikan invarian pencahayaan. Sensor kedalaman aktif (seperti *Time-of-Flight* atau *Structured Light* pada Kinect v2 atau Realsense) memancarkan sinyal inframerah mereka sendiri, sehingga bentuk geometris buah (kelengkungan permukaan bola) tetap terdeteksi bahkan dalam kegelapan atau bayangan pekat.2 Lebih jauh lagi, data kedalaman memberikan skala metrik absolut. Tanpa kedalaman, "besar" dalam gambar hanyalah masalah perspektif; buah yang dekat terlihat besar, buah yang jauh terlihat kecil. Dengan peta kedalaman yang terkalibrasi, setiap piksel memiliki koordinat $Z$ (jarak), yang memungkinkan konversi ukuran piksel ke milimeter nyata melalui prinsip *pinhole camera model*.3

## 3. Spesifikasi Data dan Protokol Akuisisi

Desain sistem ini didasarkan pada karakteristik dataset standar industri seperti **KFuji RGB-DS** dan **AmodalAppleSize\_RGB-D**. Memahami struktur *low-level* dari data ini sangat penting untuk merancang *data loader* dan lapisan input jaringan.

### 3.1. Struktur dan Format Dataset

Sistem dirancang untuk menelan data dengan spesifikasi sebagai berikut, merujuk pada standar dataset KFuji 4:

| **Atribut** | **Spesifikasi Teknis** | **Keterangan** |
| --- | --- | --- |
| **Resolusi Citra** | $548 \times 373$ piksel | Resolusi asli sensor Kinect v2 setelah registrasi. |
| **Kanal Warna (RGB)** | 8-bit Unsigned Integer (0-255) | Format standar JPEG/PNG. |
| **Kanal Kedalaman (D)** | 16-bit Unsigned Integer / 32-bit Float | Menyimpan jarak dalam milimeter (mm). Rentang tipikal 500mm - 4500mm. |
| **Kanal Sinyal (S)** | Range-Corrected IR Intensity | Intensitas inframerah yang dinormalisasi terhadap jarak, berguna untuk fitur tekstur IR. |
| **Anotasi Modal** | Poligon / Masker Biner | Area buah yang terlihat secara fisik. |
| **Anotasi Amodal** | Poligon / Masker Biner | Area estimasi buah secara keseluruhan (termasuk bagian di balik daun). |

### 3.2. Tantangan Representasi Data Kedalaman

Data kedalaman mentah dari sensor *consumer-grade* sering kali memiliki cacat yang disebut "lubang" (*holes*). Lubang ini adalah piksel dengan nilai kedalaman nol atau tidak valid, yang disebabkan oleh beberapa faktor fisik:

1. **Absorpsi IR:** Permukaan gelap atau miring menyerap sinyal inframerah.
2. **Oklusi Geometris:** Pada tepi objek, terdapat area yang terlihat oleh kamera RGB tetapi terhalang dari pandangan proyektor IR (fenomena *parallax shadow*).
3. **Specular Reflection:** Permukaan mengkilap memantulkan sinyal menjauhi sensor.

Mengirimkan peta kedalaman yang penuh lubang ini secara langsung ke dalam *Convolutional Neural Network* (CNN) sangat tidak disarankan. Lubang-lubang ini menciptakan transisi nilai ekstrem (misal: dari 2000mm ke 0mm lalu kembali ke 2000mm) yang akan diinterpretasikan oleh filter konvolusi sebagai tepi (*edge*) yang sangat kuat namun palsu. Ini menghasilkan *noise* frekuensi tinggi pada *feature map* dan menurunkan akurasi deteksi.6 Oleh karena itu, tahap *preprocessing* sinyal kedalaman menjadi mandatori.

## 4. Metodologi Pemrosesan Sinyal Kedalaman (Low-Level Preprocessing)

Kami mengusulkan penerapan filter **Structure-Aided Domain Transform Smoothing** untuk mengisi lubang dan menghaluskan peta kedalaman sebelum masuk ke jaringan saraf. Metode ini dipilih karena kemampuannya menjaga ketajaman tepi (*edge-preserving*) jauh lebih baik daripada Gaussian Blur standar, dan lebih efisien secara komputasi dibandingkan Bilateral Filter konvensional.7

### 4.1. Formulasi Matematika Domain Transform

Domain transform bekerja dengan mengubah jarak antara dua piksel. Dalam domain spasial biasa, jarak antara piksel $x$ dan $x+1$ adalah konstan. Dalam domain tertransformasi, jarak ini diperpanjang jika terdapat perbedaan warna yang signifikan antara kedua piksel tersebut (menandakan adanya tepi).

Transformasi domain didefinisikan sebagai isometri $ct: \Omega \rightarrow \mathbb{R}^d$. Jarak antara dua titik tetangga $u$ dan $v$ dalam citra didefinisikan ulang. Turunan transformasi $t$ terhadap koordinat $u$ adalah:

$$\frac{dt}{du} = 1 + \frac{\sigma\_s}{\sigma\_r} \sum\_{k=1}^{c} |\nabla I\_k(u)|$$

Di mana $\sigma\_s$ adalah parameter skala spasial, $\sigma\_r$ adalah parameter jangkauan (range), dan $\nabla I\_k$ adalah gradien citra panduan (RGB) pada kanal ke-$k$.

### 4.2. Integrasi Constraint Hibrida

Untuk aplikasi pengisian lubang pada kedalaman buah, kita memodifikasi fungsi jarak tersebut dengan tiga *constraint* (kendala) spesifik 6:

1. Struktur dan Koreksi Kedalaman ($ct\_1$):  
   Fungsi ini memastikan bahwa perambatan nilai kedalaman (smoothing) berhenti ketika bertemu dengan tepi yang kuat pada citra RGB. Ini mencegah kedalaman buah "bocor" ke latar belakang atau sebaliknya.  
     
   $$ct\_1(u) = 1 + \lambda\_1 \cdot |\nabla I(u)|$$
2. Pembobotan Saliency Visual ($ct\_2$):  
   Kita memprioritaskan perbaikan pada area yang menarik perhatian (buah) dibandingkan latar belakang. Peta saliency $S(u)$ dihitung (misalnya menggunakan metode Spectral Residual). Area lubang pada wilayah salient diberikan prioritas pengisian lebih tinggi.  
     
   $$ct\_2(u) = S(u) \cdot M\_{hole}(u)$$  
     
   Di mana $M\_{hole}$ adalah masker biner (1 jika piksel adalah lubang, 0 jika valid).
3. Lokalisasi Penghalusan Adaptif ($ct\_3$):  
   Untuk mencegah over-smoothing pada area yang datanya sudah valid dan memiliki detail halus, faktor ini mengurangi kekuatan filter pada area non-lubang.  
     
   $$ct\_3(u) = \frac{1}{1 + \exp(-\alpha \cdot (D(u) - \mu\_D))}$$

### 4.3. Implementasi Filter Rekursif

Proses penyaringan dilakukan menggunakan operasi rekursif 1D (Horizontal kemudian Vertikal) yang sangat cepat. Persamaan rekursif untuk sinyal output $J$ pada piksel ke-$i$ dengan input $f$ adalah:

$$J\_i = (1 - a\_i) f\_i + a\_i J\_{i-1}$$

Koefisien umpan balik $a\_i \in $ ditentukan oleh jarak dalam domain tertransformasi yang telah memperhitungkan ketiga constraint di atas:

$$a\_i = \exp\left(-\frac{\sqrt{2}}{\sigma\_H} (ct\_1(i) + ct\_2(i) + ct\_3(i))\right)$$

**Alur Algoritma:**

1. **Input:** Citra RGB $I$, Peta Kedalaman Mentah $D\_{raw}$.
2. **Identifikasi Lubang:** Buat masker $M\_{hole} = (D\_{raw} == 0)$.
3. **Hitung Peta Pemandu:** Hitung gradien RGB $\nabla I$ dan peta Saliency $S$.
4. **Hitung Koefisien Transmisi:** Hitung $a\_i$ untuk setiap piksel berdasarkan $ct\_1, ct\_2, ct\_3$.
5. **Pass Horizontal:** Lakukan filter rekursif kiri-ke-kanan dan kanan-ke-kiri.
6. **Pass Vertikal:** Lakukan filter rekursif atas-ke-bawah dan bawah-ke-atas pada hasil pass horizontal.
7. **Refinement:** Gabungkan hasil filter $D\_{filtered}$ dengan data asli. Untuk piksel valid ($M\_{hole}=0$), kita bisa memilih mempertahankan nilai asli $D\_{raw}$ atau menggunakan campuran terbobot untuk *denoising*. Untuk piksel lubang ($M\_{hole}=1$), gunakan nilai $D\_{filtered}$.

## 5. Arsitektur Jaringan Neural: Modifikasi YOLOv8 untuk Input 4-Channel dan Output Amodal

Inti dari proposal ini adalah modifikasi struktural pada YOLOv8. YOLOv8 dipilih karena keseimbangannya antara kecepatan (*real-time inference*) dan akurasi, serta fleksibilitas arsitekturnya yang berbasis *anchor-free* dan *decoupled head*.9

### 5.1. Modifikasi Input Backbone (4-Channel Integration)

Model standar YOLOv8 dilatih pada dataset COCO yang berisi citra RGB 3-channel. Lapisan konvolusi pertama (*stem layer*) dirancang untuk menerima tensor berukuran $(Batch, 3, Height, Width)$. Kita perlu mengubah ini menjadi $(Batch, 4, Height, Width)$ untuk mengakomodasi kanal Depth yang telah diproses.10

#### 5.1.1. Konfigurasi YAML

Langkah pertama adalah membuat file konfigurasi model kustom, misalnya yolov8-rgbd-amodal.yaml. Dalam file ini, parameter ch (channel) harus dieksplisitkan.

YAML

# yolov8-rgbd-amodal.yaml  
nc: 1 # number of classes (Apple)  
ch: 4 # Input channels (RGB + Depth)  
  
# Backbone  
backbone:  
 # [from, repeats, module, args]  
 - [-1, 1, Conv, ] # Layer 0: Conv2d(in\_ch=4, out=64, k=3, s=2)  
 - [-1, 1, Conv, ] # Layer 1  
 -] # Layer 2  
 ...

Perubahan kritis terjadi pada mekanisme *parsing* model. Dalam pustaka Ultralytics, fungsi parse\_model di ultralytics/nn/tasks.py membaca argumen ch. Jika kita menyuplai ch=4 saat inisialisasi model, argumen ini akan diteruskan ke konstruktor modul Conv pertama.

#### 5.1.2. Strategi Inisialisasi Bobot (Mengatasi Dimensi Mismatch)

Kita tidak bisa sekadar memuat bobot prate-latih (*pretrained weights*) yolov8n.pt ke model baru ini karena ketidakcocokan dimensi pada layer pertama. Bobot asli memiliki bentuk $(64, 3, 3, 3)$, sedangkan layer baru membutuhkan $(64, 4, 3, 3)$.

Strategi yang diusulkan adalah **Transfer Learning dengan Zero-Initialization pada Kanal Tambahan**.12

1. Salin Bobot RGB: Bobot untuk 3 kanal pertama (RGB) disalin langsung dari model pra-latih. Ini mempertahankan kemampuan ekstraksi fitur visual dasar (tepi, warna, tekstur).  
     
   $$W\_{new}[:, 0:3, :, :] = W\_{pretrained}$$
2. Inisialisasi Kanal Depth: Kanal ke-4 diinisialisasi dengan nol atau nilai mendekati nol.  
     
   $$W\_{new}[:, 3, :, :] = 0$$  
     
   Alasan: Dengan inisialisasi nol, pada iterasi pertama pelatihan, kontribusi kanal kedalaman terhadap output fitur adalah nol. Model akan berperilaku persis seperti model RGB standar. Seiring berjalannya backpropagation, gradien akan mulai mengalir ke kanal kedalaman, memungkinkan jaringan untuk perlahan-lahan mempelajari korelasi antara kedalaman dan fitur buah tanpa merusak fitur visual yang sudah mapan (avoiding feature shock).

### 5.2. Desain Head Segmentasi Amodal (Dual-Path ProtoNet)

YOLOv8 menggunakan pendekatan segmentasi berbasis YOLACT (*You Only Look At CoefficienTs*), yang memisahkan tugas menjadi dua jalur paralel 13:

1. **ProtoNet:** Jaringan Fully Convolutional yang memprediksi serangkaian "masker prototipe" ($P$) yang independen terhadap instans spesifik. Bentuk tensor: $(Batch, 32, H/4, W/4)$.
2. **Mask Coefficients:** Head deteksi memprediksi vektor koefisien ($C$) untuk setiap *anchor* atau *proposal*. Bentuk vektor: $32$ elemen per deteksi.

Masker final $M$ dihasilkan melalui operasi perkalian matriks linear diikuti fungsi aktivasi sigmoid:

$$M = \sigma(P \times C^T)$$

Untuk **Segmentasi Amodal**, kita membutuhkan *dua* output masker per deteksi: satu untuk bagian terlihat (*visible*) dan satu untuk bentuk utuh (*amodal*).

Modifikasi yang Diusulkan:

Alih-alih membuat dua ProtoNet (yang mahal secara komputasi), kita akan memperluas vektor koefisien. ProtoNet akan belajar basis fitur yang cukup kaya untuk merekonstruksi kedua jenis masker tersebut.

1. **Ekspansi Koefisien:** Head deteksi dimodifikasi untuk memprediksi $2 \times k$ koefisien, di mana $k=32$ (jumlah prototipe).
   * $C\_{vis}$ (32 elemen pertama): Koefisien untuk masker visible.
   * $C\_{amodal}$ (32 elemen kedua): Koefisien untuk masker amodal.
2. Assembly Masker:  
     
   $$M\_{vis} = \sigma(P \times C\_{vis}^T)$$  
   $$M\_{amodal} = \sigma(P \times C\_{amodal}^T)$$

Ini memungkinkan jaringan untuk menggunakan basis visual yang sama (misalnya, tepi melengkung) untuk kedua tugas, namun memberikan bobot yang berbeda. Misalnya, prototipe yang merepresentasikan "tepi bawah" mungkin berbobot tinggi untuk $M\_{vis}$ dan $M\_{amodal}$, sedangkan prototipe "tepi atas" yang tertutup daun mungkin memiliki bobot negatif di $M\_{vis}$ tetapi positif di $M\_{amodal}$.

### 5.3. Fungsi Kerugian (Loss Function)

Fungsi kerugian total harus mengakomodasi kedua tugas segmentasi:

$$L\_{total} = \lambda\_{box}L\_{box} + \lambda\_{cls}L\_{cls} + \lambda\_{dfl}L\_{dfl} + \lambda\_{seg}(L\_{mask}^{vis} + L\_{mask}^{amodal})$$

Di mana $L\_{mask}$ biasanya adalah kombinasi dari Binary Cross Entropy (BCE) dan Dice Loss untuk memastikan akurasi piksel dan overlap yang baik. Penambahan $L\_{mask}^{amodal}$ memaksa jaringan untuk mempelajari bentuk di balik oklusi berdasarkan ground truth amodal dari dataset.

## 6. Pipeline Pelacakan dan Lokalisasi 3D

Setelah model memprediksi masker amodal dan visible, langkah selanjutnya adalah mengubah informasi ini menjadi data 3D metrik yang dapat dilacak (*trackable*) dan diukur.

### 6.1. Ekstraksi Centroid 3D dan Penanganan Oklusi

Metode pelacakan konvensional (seperti DeepSORT) bekerja pada bidang gambar 2D. Namun, pelacakan 2D gagal saat kamera bergerak maju-mundur (zoom effect) atau saat oklusi terjadi. Kami mengusulkan pelacakan berbasis **Centroid 3D**.

1. **Masking Kedalaman:** Gunakan $M\_{vis}$ (masker visible) untuk mengambil nilai kedalaman dari peta kedalaman $D$. *Penting:* Jangan gunakan $M\_{amodal}$ untuk mengambil kedalaman, karena $M\_{amodal}$ mencakup piksel daun/penghalang yang memiliki nilai kedalaman berbeda (lebih dekat).
2. Estimasi Jarak ($Z\_c$): Hitung nilai median dari piksel kedalaman dalam area $M\_{vis}$. Median lebih kuat (robust) terhadap noise outlier dibandingkan rata-rata (mean).  
     
   $$Z\_c = \text{median}(D[M\_{vis} == 1])$$
3. **Kalkulasi Centroid 2D ($u\_c, v\_c$):** Hitung titik pusat massa dari $M\_{amodal}$ (masker amodal). Ini memberikan pusat geometris buah yang sebenarnya, bukan pusat dari bagian yang terlihat saja.

### 6.2. Back-Projection (Transformasi 2D ke 3D)

Dengan asumsi model kamera *pinhole* dan parameter intrinsik kamera (Focal Length $f\_x, f\_y$ dan Principal Point $c\_x, c\_y$) diketahui dari kalibrasi, kita proyeksikan titik pusat $(u\_c, v\_c)$ dengan kedalaman $Z\_c$ ke koordinat dunia 3D $(X, Y, Z)$.15

Persamaan proyeksi balik:

$$X = \frac{(u\_c - c\_x) \cdot Z\_c}{f\_x}$$

$$Y = \frac{(v\_c - c\_y) \cdot Z\_c}{f\_y}$$

$$Z = Z\_c$$

Hasilnya adalah himpunan titik 3D $\mathcal{O}\_t = \{P\_1, P\_2, \dots, P\_n\}$ untuk frame saat ini $t$, di mana setiap titik merepresentasikan lokasi fisik buah dalam ruang metrik (misalnya milimeter relatif terhadap kamera).

### 6.3. Algoritma Pencocokan Hungarian 3D

Untuk melacak buah antar frame, kita menggunakan algoritma pencocokan bipartit (Hungarian Algorithm) dengan matriks biaya berbasis jarak Euclidean 3D.

1. **Prediksi State:** Untuk setiap trek yang ada, gunakan **Kalman Filter** untuk memprediksi posisi 3D-nya di frame saat ini berdasarkan kecepatan sebelumnya. Ini penting untuk menangani buah yang hilang sesaat.
2. Matriks Biaya ($C$): Hitung jarak antara prediksi trek $T\_i$ dan deteksi baru $D\_j$.  
     
   $$C\_{ij} = \sqrt{(X\_{T\_i} - X\_{D\_j})^2 + (Y\_{T\_i} - Y\_{D\_j})^2 + (Z\_{T\_i} - Z\_{D\_j})^2}$$  
     
   Penggunaan jarak 3D jauh lebih akurat daripada IoU (Intersection over Union) 2D karena buah yang berdekatan secara visual di gambar (satu di depan yang lain) akan terpisah jauh dalam sumbu Z.17
3. **Assignment:** Minimalkan total biaya menggunakan algoritma Hungarian. Terapkan ambang batas (*gating threshold*) jarak maksimum (misal: 100mm). Jika jarak > threshold, jangan pasangkan.
4. **Manajemen Trek:**
   * **Unmatched Detection:** Buat trek baru (potensi buah baru masuk frame).
   * **Unmatched Track:** Tingkatkan counter lost\_frames. Jika lost\_frames > max\_age, hapus trek (buah keluar frame).
   * **Matched:** Update posisi Kalman Filter dengan pengukuran baru.

### 6.4. Logika Anti-Double Counting

Masalah utama dalam penghitungan buah adalah buah yang terhalang total selama beberapa frame lalu muncul kembali. Sistem 2D akan menganggapnya buah baru.

Dengan sistem 3D ini, jika buah terhalang total, treknya masuk status "Lost" tetapi posisinya diprediksi oleh Kalman Filter. Saat buah muncul kembali, posisinya di 3D akan sangat dekat dengan prediksi trek yang "hilang" tersebut (karena buah statis relatif terhadap pohon), sehingga sistem akan menyambungkan kembali ID yang sama, mencegah penghitungan ganda.18

## 7. Rencana Implementasi Bertahap

Berikut adalah *roadmap* teknis untuk mengimplementasikan sistem ini dari nol.

### Tahap 1: Persiapan Lingkungan dan Data

1. **Instalasi Dependensi:** PyTorch, Ultralytics, Open3D (visualisasi point cloud), Albumentations (augmentasi).
2. **Dataset Preprocessing:**
   * Unduh KFuji RGB-DS dataset.
   * Konversi format .mat (Matlab) ke format tensor .npy atau .tiff yang mendukung 4 channel.
   * Terapkan *Global Normalization* pada data kedalaman: $D\_{norm} = (D - D\_{min}) / (D\_{max} - D\_{min})$.
   * Siapkan struktur folder YOLO standar (images/train, labels/train), tetapi file gambar berisi 4 channel.

### Tahap 2: Kustomisasi Model dan Training

1. **Modifikasi Dataloader:** Override kelas LoadImages di Ultralytics untuk mendukung pembacaan file .npy 4-channel.
2. **Definisi Model:** Buat file yolov8-custom.yaml dengan ch: 4.
3. **Script Training:**
   * Muat model: model = YOLO('yolov8-custom.yaml').
   * Transfer bobot: Muat state\_dict dari yolov8n-seg.pt. Salin bobot layer 0 ke tensor baru, inisialisasi bobot kanal ke-4 dengan nol.
   * Train model dengan parameter: epochs=300, imgsz=640, batch=16.
   * Gunakan teknik **Freezing**: Bekukan 4 layer pertama backbone (model.0 sampai model.4) untuk mempertahankan ekstraktor fitur dasar dan mempercepat konvergensi pada dataset kecil.20

### Tahap 3: Pengembangan Pipeline Pelacakan

1. **Kalibrasi Kamera:** Pastikan parameter intrinsik ($f\_x, f\_y, c\_x, c\_y$) dari sensor Kinect v2/Realsense tersimpan dalam file konfigurasi.
2. **Implementasi Modul Tracking:** Tulis kelas Python Tracker3D yang membungkus filter Kalman dan algoritma Hungarian (menggunakan scipy.optimize.linear\_sum\_assignment).
3. **Integrasi:** Buat skrip inferensi utama yang:
   * Membaca frame video RGB-D.
   * Menjalankan inferensi YOLO.
   * Melakukan back-projection.
   * Mengupdate tracker.
   * Memvisualisasikan hasil (Bounding Box + Masker Amodal + Estimasi Diameter).

## 8. Analisis Kinerja dan Metrik Evaluasi

Keberhasilan sistem akan diukur menggunakan metrik kuantitatif yang ketat:

| **Metrik** | **Deskripsi** | **Target Kinerja** |
| --- | --- | --- |
| **mAP@0.5 (Box & Mask)** | Rata-rata presisi deteksi dan segmentasi amodal. | $> 90\%$ (mengacu pada SOTA deteksi apel) |
| **Counting Error Rate** | Persentase kesalahan hitung total per baris pohon. $\frac{\|Count\_{est} - Count\_{GT}\|}{Count\_{GT$ | $< 5\%$ |
| **Sizing RMSE** | Akar kuadrat rata-rata kesalahan estimasi diameter (mm) dibandingkan pengukuran caliper manual. | $< 3.5$ mm |
| **ID Switch** | Jumlah kali identitas objek tertukar selama pelacakan. | Serendah mungkin (mengindikasikan stabilitas pelacakan). |

## 9. Kesimpulan

Proposal ini menyajikan pendekatan *low-level* yang radikal namun terstruktur untuk memecahkan masalah klasik dalam visi komputer pertanian. Dengan memodifikasi inti arsitektur YOLOv8 untuk menerima persepsi kedalaman secara natif dan memprediksi bentuk amodal, kita menghilangkan ambiguitas visual yang disebabkan oleh oklusi. Lebih jauh lagi, dengan mengangkat logika pelacakan dari bidang gambar 2D ke ruang metrik 3D, sistem ini menjanjikan robustas yang jauh lebih tinggi terhadap pergerakan kamera dan gangguan lingkungan dinamis. Implementasi sistem ini tidak hanya akan menghasilkan penghitungan yang lebih akurat, tetapi juga data fenotipe (ukuran buah) yang *actionable* bagi petani untuk manajemen panen presisi.

**Daftar Tabel Referensi Teknis:**

* **Tabel 1:** Spesifikasi Format Data KFuji RGB-DS.
* **Tabel 2:** Parameter Filter Domain Transform ($ \sigma\_s, \sigma\_r, \lambda\_1 $).
* **Tabel 3:** Struktur Tensor Modifikasi Layer Input YOLOv8.
* **Tabel 4:** Target Kinerja Evaluasi Sistem.

#### Works cited

1. Amodal Segmentation Based on Visible Region Segmentation and Shape Prior, accessed December 12, 2025, <https://cdn.aaai.org/ojs/16407/16407-13-19901-1-2-20210518.pdf>
2. [1804.00307] Robust Fruit Counting: Combining Deep Learning, Tracking, and Structure from Motion - arXiv, accessed December 12, 2025, <https://arxiv.org/abs/1804.00307>
3. GRAP-UdL-AT/Amodal\_Fruit\_Sizing: Fruit detection and measurment using modal and amodal instance segmentation in RGB-D images - GitHub, accessed December 12, 2025, <https://github.com/GRAP-UdL-AT/Amodal_Fruit_Sizing>
4. Multi-modal deep learning for Fuji apple detection using RGB-D cameras and their radiometric capabilities | Request PDF - ResearchGate, accessed December 12, 2025, <https://www.researchgate.net/publication/333210370_Multi-modal_deep_learning_for_Fuji_apple_detection_using_RGB-D_cameras_and_their_radiometric_capabilities>
5. KFuji RGB-DS database: Fuji apple multi-modal ... - UPCommons, accessed December 12, 2025, <https://upcommons.upc.edu/bitstreams/16bac283-e20d-470c-83d3-425eca3aef06/download>
6. An efficient depth map preprocessing method based on structure-aided domain transform smoothing for 3D view generation | PLOS One, accessed December 12, 2025, <https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0175910>
7. An efficient depth map preprocessing method based on structure-aided domain transform smoothing for 3D view generation - ResearchGate, accessed December 12, 2025, <https://www.researchgate.net/publication/316116423_An_efficient_depth_map_preprocessing_method_based_on_structure-aided_domain_transform_smoothing_for_3D_view_generation>
8. An efficient depth map preprocessing method based on structure ..., accessed December 12, 2025, <https://pmc.ncbi.nlm.nih.gov/articles/PMC5391110/>
9. Algorithm principles and implementation with YOLOv8 - MMYOLO's documentation!, accessed December 12, 2025, <https://mmyolo.readthedocs.io/en/latest/recommended_topics/algorithm_descriptions/yolov8_description.html>
10. How to make changes in YOLOv8 architecture as a total beginner. · ultralytics · Discussion #10211 - GitHub, accessed December 12, 2025, <https://github.com/orgs/ultralytics/discussions/10211>
11. RGBD Input for YOLOv8 trying to pass 4 channels · Issue #16024 - GitHub, accessed December 12, 2025, <https://github.com/ultralytics/ultralytics/issues/16024>
12. Segmentation with 4 channels : r/computervision - Reddit, accessed December 12, 2025, <https://www.reddit.com/r/computervision/comments/1agtav2/segmentation_with_4_channels/>
13. summary of YOLOv8-Seg model structure · Issue #1710 - GitHub, accessed December 12, 2025, <https://github.com/ultralytics/ultralytics/issues/1710>
14. How to implement instance segmentation using YOLOv8 neural network - DEV Community, accessed December 12, 2025, <https://dev.to/andreygermanov/how-to-implement-instance-segmentation-using-yolov8-neural-network-3if9>
15. Camera Calibration and 3D Reconstruction - OpenCV Documentation, accessed December 12, 2025, <https://docs.opencv.org/4.x/d9/d0c/group__calib3d.html>
16. Camera model: intrinsic parameters - Hoàng-Ân Lê, accessed December 12, 2025, <https://lhoangan.github.io/camera-params/>
17. Tracking and Counting Apples in Orchards Under Intermittent Occlusions and Low Frame Rates - ResearchGate, accessed December 12, 2025, <https://www.researchgate.net/publication/381861188_Tracking_and_Counting_Apples_in_Orchards_Under_Intermittent_Occlusions_and_Low_Frame_Rates>
18. Automatic Apple Detection and Counting with AD-YOLO and MR-SORT - ResearchGate, accessed December 12, 2025, <https://www.researchgate.net/publication/385478664_Automatic_Apple_Detection_and_Counting_with_AD-YOLO_and_MR-SORT>
19. Fruit detection and 3D location using instance segmentation neural networks and structure-from-motion photogrammetry - UPCommons, accessed December 12, 2025, <https://upcommons.upc.edu/bitstreams/ab54296b-7132-4b90-ad76-b822e6654e62/download>
20. An Analysis of Layer-Freezing Strategies for Enhanced Transfer Learning in YOLO Architectures - ResearchGate, accessed December 12, 2025, <https://www.researchgate.net/publication/394470267_An_Analysis_of_Layer-Freezing_Strategies_for_Enhanced_Transfer_Learning_in_YOLO_Architectures>