Optimasi Sudut Kamera dalam Fotografi 3D Menggunakan Algoritma *Greedy Best-First Search* dengan Simulasi pada Blender3D

Barru Adi Utomo - 13523101
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail: barru.adi@gmail.com , 13523101@std.stei.itb.ac.id

Abstrak—Penentuan sudut pandang kamera yang optimal merupakan salah satu faktor krusial dalam proses fotogrametri untuk rekonstruksi objek 3D. Proses manual seringkali tidak efisien, memakan waktu, dan subjektif, sehingga berpotensi menghasilkan model 3D yang tidak lengkap atau berkualitas rendah. Penelitian ini mengusulkan sebuah metode untuk mengoptimasi penempatan kamera secara otomatis menggunakan algoritma *Best-First Search*. Lingkungan simulasi dibangun menggunakan perangkat lunak Blender, yang memungkinkan pengujian metode pada berbagai model 3D virtual. Algoritma ini bekerja dengan mengevaluasi setiap kemungkinan posisi kamera berdasarkan fungsi heuristik yang dirancang untuk memaksimalkan cakupan permukaan objek dan meminimalkan oklusi. Hasil simulasi menunjukkan bahwa metode yang diusulkan mampu menentukan set posisi kamera yang efektif dan efisien. Dibandingkan dengan metode penempatan manual atau *grid-based*, pendekatan *Best-First Search* secara signifikan mengurangi jumlah kamera yang dibutuhkan sambil tetap mempertahankan atau bahkan meningkatkan kualitas cakupan area permukaan model. Metode ini memberikan kontribusi berupa solusi otomatisasi yang cerdas dalam alur kerja fotografi 3D, berpotensi meningkatkan efisiensi dan kualitas rekonstruksi digital.

Kata Kunci—Optimasi, Sudut Kamera, Fotografi 3D, Best-First Search, Blender, Simulasi, Fotogrametri.

I. PENDAHULUAN

A. Latar Belakang

Perkembangan teknologi digital telah mendorong kebutuhan akan model tiga dimensi (3D) di berbagai sektor, mulai dari industri hiburan seperti film dan permainan video, pelestarian warisan budaya (*digital heritage*), rekayasa balik (*reverse engineering*), hingga visualisasi medis. Salah satu metode yang paling populer dan ekonomis untuk menghasilkan model 3D dari objek nyata adalah fotogrametri. Teknik ini bekerja dengan cara merekonstruksi bentuk, tekstur, dan geometri suatu objek berdasarkan serangkaian foto yang diambil dari berbagai sudut pandang yang berbeda.

Kualitas hasil akhir dari sebuah model 3D yang dibuat melalui fotogrametri sangat bergantung pada kualitas dan kelengkapan data input, yaitu set foto yang digunakan. Untuk mencapai rekonstruksi yang akurat dan lengkap, kamera harus mampu menangkap seluruh permukaan objek secara detail tanpa ada bagian yang terlewat. Penentuan posisi dan sudut pandang kamera (*viewpoint*) menjadi faktor krusial dalam proses ini. Penempatan kamera yang buruk atau tidak sistematis dapat mengakibatkan model 3D yang tidak lengkap, memiliki lubang, atau mengalami distorsi geometris.

Secara konvensional, proses pengambilan gambar untuk fotogrametri sering dilakukan secara manual. Operator akan menggerakkan kamera di sekitar objek berdasarkan intuisi dan pengalaman untuk memastikan semua bagian terekam. Namun, pendekatan manual ini memiliki beberapa kelemahan signifikan bersifat subjektif, memakan waktu lama, repetitif, dan sering kali tidak efisien. Operator mungkin mengambil foto lebih dari yang dibutuhkan di beberapa area dan justru melewatkan area lain yang sulit dijangkau. Hal ini memicu kebutuhan akan sebuah sistem yang dapat mengotomatisasi proses perencanaan sudut pandang kamera secara cerdas dan optimal.

Oleh karena itu, penelitian ini mengusulkan sebuah solusi untuk mengotomatisasi penentuan sudut kamera dengan memanfaatkan algoritma pencarian cerdas. Algoritma *Best-First Search* dipilih karena kemampuannya untuk mengeksplorasi ruang pencarian secara efisien dengan memprioritaskan jalur yang paling menjanjikan berdasarkan fungsi heuristik. Untuk memvalidasi metode ini secara aman, cepat, dan hemat biaya, seluruh proses akan disimulasikan dalam lingkungan virtual menggunakan perangkat lunak Blender. Blender, dengan *Application Programming Interface* (API) Python yang kuat, memungkinkan pembuatan skenario yang terkontrol untuk menguji dan menyempurnakan algoritma optimasi ini sebelum diterapkan pada sistem robotik fisik [1].

B. Rumusan Masalah

Berdasarkan latar belakang yang telah diuraikan, maka rumusan masalah dalam penelitian ini adalah sebagai berikut:

1. Bagaimana merancang sebuah sistem otomatis untuk menentukan serangkaian sudut kamera yang optimal untuk fotografi 3D pada sebuah objek target?

2. Bagaimana mengimplementasikan algoritma *Best-First Search* untuk memandu proses pencarian sudut kamera terbaik dalam lingkungan simulasi?

C. Tujuan Penelitian

Adapun tujuan yang ingin dicapai melalui penelitian ini adalah:

1. Merancang dan membangun sebuah sistem otomatis untuk optimasi sudut kamera menggunakan algoritma *Best-First Search* di dalam lingkungan simulasi Blender.
2. Menganalisis kinerja sistem yang diusulkan berdasarkan metrik evaluasi seperti persentase cakupan permukaan objek, posisi kamera, dan skor berdasarkan perhitungan.

D. Batasan Masalah

Untuk menjaga agar penelitian ini tetap terarah, maka ditetapkan batasan-batasan masalah sebagai berikut:

1. Penelitian dilakukan sepenuhnya dalam lingkungan simulasi virtual menggunakan perangkat lunak Blender. Tidak ada implementasi yang melibatkan perangkat keras fisik.
2. Algoritma optimasi yang digunakan adalah *Greedy Best-First Search*. Penelitian ini tidak melakukan perbandingan komprehensif dengan algoritma pencarian heuristik lainnya (misalnya A*, *Greedy*, atau algoritma genetika).
3. Fungsi heuristik yang dirancang untuk memandu algoritma berfokus pada metrik geometris, yaitu memaksimalkan jumlah permukaan atau *vertex* baru yang terlihat, dan tidak mempertimbangkan faktor fotometrik seperti kualitas pencahayaan atau sifat material objek.

II. Tinjauan Pustaka

A. Fotografi 3D dan Fotogrametri

Fotogrametri adalah seni dan ilmu untuk memperoleh informasi yang andal tentang objek fisik dan lingkungan melalui proses perekaman, pengukuran, dan interpretasi citra fotografis [2]. Dalam konteks rekayasa digital, fotogrametri, khususnya teknik *Structure from Motion* (SfM), telah menjadi metode yang sangat populer untuk membuat model 3D dari objek nyata.

Prinsip kerja SfM adalah merekonstruksi geometri tiga dimensi secara simultan dari serangkaian gambar dua dimensi yang diambil dari sudut pandang yang berbeda. Proses ini secara umum melibatkan beberapa tahapan kunci:

- Ekstraksi Fitur: Mengidentifikasi titik-titik kunci yang unik pada setiap gambar.
- Pencocokan Fitur: Mencari korespondensi titik-titik kunci yang sama di antara beberapa gambar.
- Rekonstruksi Geometris: Menggunakan informasi korespondensi dan prinsip triangulasi untuk secara bersamaan menghitung posisi 3D dari titik-titik fitur dan parameter internal maupun eksternal dari setiap posisi kamera.
- Densifikasi: Setelah posisi kamera diketahui, proses *Multi-View Stereo* (MVS) digunakan untuk menghasilkan dense point cloud yang jauh lebih detail, yang kemudian diubah menjadi *mesh* poligonal.

Kualitas akhir dari model 3D sangat bergantung pada kualitas set gambar masukan. Faktor-faktor krusial termasuk tumpang tindih (*overlap*) yang cukup antar gambar, pencahayaan yang merata, dan yang paling relevan untuk penelitian ini, adalah cakupan sudut pandang yang komprehensif untuk meminimalkan bagian yang tidak terlihat (*occlusion*) dan memastikan akurasi geometris [3]. Kegagalan dalam menyediakan data visual yang lengkap akan menghasilkan model 3D yang berlubang, tidak akurat, atau bahkan gagal sama sekali.

B. Masalah Next-Best-View

Tantangan untuk secara otomatis menentukan di mana kamera harus diposisikan selanjutnya untuk mendapatkan informasi paling berguna dikenal dalam literatur robotika dan visi komputer sebagai masalah *Next-Best-View* (NBV). Masalah ini pertama kali diformalkan oleh Connolly (1985) [4] dan sejak saat itu menjadi area penelitian yang aktif.

Tujuan utama dari algoritma NBV adalah untuk membangun model 3D yang lengkap dan akurat dari sebuah objek dengan jumlah pandangan (*views*) yang seminimal mungkin. Pendekatan ini secara inheren lebih efisien daripada metode akuisisi data non-adaptif, seperti memindai dari grid atau jalur yang telah ditentukan sebelumnya. Sebuah sistem NBV yang efektif secara iteratif menjawab pertanyaan: "Berdasarkan informasi yang telah saya kumpulkan sejauh ini, ke mana saya harus melihat selanjutnya untuk memaksimalkan perolehan informasi baru?"

Solusi untuk masalah NBV umumnya memerlukan tiga komponen utama:

- Representasi Model: Model objek yang terus diperbarui seiring dengan masuknya data baru.
- Pemilihan Kandidat Pandangan: Menentukan set posisi dan orientasi kamera yang mungkin untuk dievaluasi.
- Fungsi Utilitas (Heuristik): Sebuah metrik untuk mengukur nilai dari setiap kandidat pandangan. Metrik umum termasuk jumlah *voxels* batas yang terlihat, luas permukaan yang diproyeksikan, atau pengurangan entropi model [5][6].

Penelitian ini secara langsung mengatasi masalah NBV dengan menggunakan algoritma pencarian heuristik untuk memilih pandangan berikutnya secara cerdas.

C. Algoritma Best First Search

Best-First Search adalah sebuah algoritma pencarian yang mengeksplorasi graf dengan cara memperluas simpul (*node*) yang paling menjanjikan, yang dipilih berdasarkan aturan atau fungsi evaluasi tertentu [7]. Algoritma ini tergolong dalam

kategori pencarian terinformasi (*informed search*) karena menggunakan fungsi heuristik untuk memandu proses pencariannya, membuatnya berpotensi jauh lebih efisien daripada pencarian tak terinformasi (*uninformed search*) seperti *Breadth-First Search* atau Depth-*First Search*.

Mekanisme kerja *Best-First Search* melibatkan dua struktur data utama:

- Open List

  Sebuah antrean prioritas (*priority queue*) yang menyimpan semua simpul yang telah ditemukan tetapi belum dievaluasi. Simpul dengan nilai heuristik terbaik (misalnya, skor tertinggi) memiliki prioritas tertinggi untuk dieksplorasi.

- Closed List

  Sebuah set yang menyimpan semua simpul yang telah dievaluasi, untuk mencegah algoritma mengunjungi kembali simpul yang sama dan terperangkap dalam siklus tak terbatas.

Dalam konteks masalah NBV, *Best-First Search* dapat diterapkan dengan pemetaan sebagai berikut:

- Simpul (*Node*): Setiap kandidat posisi kamera.
- Fungsi Evaluasi (Heuristik): Fungsi utilitas NBV, yaitu skor yang mengukur seberapa baik sebuah pandangan.
- Tujuan: Mencapai kondisi di mana model dianggap selesai sesuai kriteria berhenti.

Bentuk paling umum dari *Best-First Search* adalah *Greedy Best-First Search*, yang secara serakah selalu memilih simpul yang tampak terbaik secara lokal. Meskipun tidak menjamin penemuan jalur terpendek seperti algoritma A*, pendekatan ini sangat efektif untuk masalah di mana biaya jalur tidak relevan dan tujuannya adalah untuk mencapai sebuah *goal state* secepat mungkin [8].

D. Blender sebagai Simulasi

Blender adalah *software* pemodelan 3D gratis dan *open-source* yang dilengkapi dengan fitur lengkap untuk pemodelan, animasi, rendering, dan komposisi. Untuk penelitian di bidang robotika dan visi komputer, Blender mempunyai beberapa keunggulan signifikan sebagai platform simulasi [1]:

- Python API (bpy)

  Blender menyediakan antarmuka pemrograman aplikasi (API) Python yang komprehensif, memungkinkan kontrol penuh atas hampir setiap aspek di dalam scene. Peneliti dapat secara terprogram membuat dan memanipulasi objek, mengatur posisi dan orientasi kamera, mengubah material, serta mengontrol mesin render. Ini adalah tulang punggung dari implementasi algoritma dalam penelitian ini.

- Mesin *Render* dan *Ray-Casting*

  Blender memiliki mesin render fotorealistis (*Cycles*) dan real-time (*Eevee*). Yang lebih penting, API-nya menyediakan akses langsung ke fungsi *ray-casting* (`scene.ray_cast`). Fungsi ini memungkinkan simulasi garis pandang kamera secara akurat dan efisien untuk menentukan visibilitas permukaan objek, yang merupakan inti dari perhitungan fungsi heuristik NBV.

- Akurasi Fisik dan Visual

  Lingkungan simulasi memungkinkan pengujian yang cepat, dapat diulang, dan berbiaya rendah. Algoritma dapat diuji pada berbagai model 3D dengan kompleksitas yang berbeda dalam kondisi yang terkontrol sempurna, sebelum diimplementasikan pada perangkat keras fisik yang mahal dan berisiko.

- Visualisasi Intuitif

  Proses pengambilan keputusan algoritma, seperti jalur kamera yang dipilih, dapat divisualisasikan secara langsung di 3D *Viewport*, mempermudah proses *debugging*, analisis, dan presentasi hasil penelitian.

III. METODOLOGI PENELITIAN

A. Kerangka Penelitian

Penelitian ini dilaksanakan melalui beberapa tahapan yang terstruktur. Tahapan-tahapan tersebut adalah sebagai berikut:

1. Studi Literatur

   Mengkaji teori-teori dasar mengenai fotogrametri, masalah *Next-Best-View* (NBV), algoritma *Best-First Search*, serta penelitian terkait sebelumnya untuk membangun landasan teoretis yang kuat.

2. Perancangan Sistem

   Merancang arsitektur sistem secara keseluruhan, termasuk penentuan lingkungan simulasi, representasi ruang pencarian, dan desain fungsi heuristik gabungan.

3. Implementasi

   Menerjemahkan rancangan sistem ke dalam kode program menggunakan Python API (`bpy`) di dalam perangkat lunak Blender.

4. Pengujian

   Menjalankan serangkaian eksperimen berdasarkan skenario pengujian yang telah ditentukan untuk mengumpulkan data kinerja sistem.

5. Analisis Data

   Menganalisis data hasil pengujian dengan membandingkan performa metode yang diusulkan terhadap metode pembanding menggunakan metrik evaluasi yang telah ditetapkan.

6. Penarikan Kesimpulan

   Menyimpulkan hasil analisis dan memberikan saran untuk pengembangan penelitian di masa depan.

B. *Objek Penelitian*

Objek yang digunakan dalam pengujian adalah model 3D standar yang umum digunakan dalam grafika komputer untuk memastikan hasil dapat dibandingkan. Objek utama adalah Primitif Monyet "Suzanne". Objek tersebut dipilih karena memiliki geometri yang cukup kompleks, mencakup area cembung, cekung, dan detail-detail halus yang dapat menyebabkan oklusi, sehingga menjadi kasus uji yang baik untuk algoritma perencanaan pandangan.

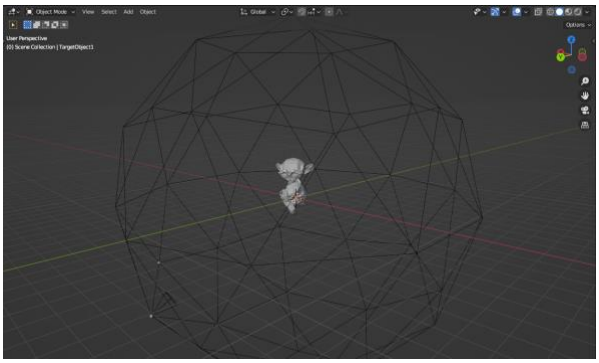
C. *Prosedur Penelitian*

Prosedur penelitian dibagi menjadi beberapa tahap implementasi dan pengujian.

1) *Persiapan Lingkungan Simulasi*

Lingkungan simulasi disiapkan di dalam Blender dengan langkah-langkah berikut:

- Objek target ditempatkan sesuai keinginan percobaan dan dalam jumlah yang dibebaskan. Nama objek ini diganti menjadi `TargetObject`.
- Sebuah objek Icosphere ditambahkan ke scene dan diskalakan hingga mengelilingi semua `TargetObject` sepenuhnya. Objek ini berfungsi sebagai sumber kandidat posisi kamera dan nama objek diganti menjadi `ViewSphere`.
- Sebuah objek `Camera` dipastikan ada di dalam scene.



Gambar 1. Objek pada *Viewport* Blender

2) *Perancangan Sistem dan Algoritma*

Algoritma pencarian diimplementasikan dalam bahasa Python skrip di dalam Blender3D.

a) *Representasi Ruang Pencarian*

Ruang pencarian, atau himpunan semua kemungkinan posisi kamera, direpresentasikan secara diskrit. Setiap vertex pada objek `ViewSphere` dianggap sebagai satu kandidat posisi kamera. Pendekatan ini menyederhanakan masalah pencarian kontinu menjadi pencarian pada graf diskrit.

Terdapat konfigurasi untuk kode yang dapat diubah sesuai bobot yang diinginkan. Berikut merupakan konfigurasi yang ada dalam program

```
CONFIG = {
    "TARGET_OBJECT_NAMES": ["TargetObject"],
    "VIEWSPHERE_NAME": "ViewSphere",
    "CAMERA_NAME": "Camera",

    "W_COVERAGE": 0.5,
    "W_ROT": 0.35,
    "W_HEADROOM": 0.15,

    "TARGET_COVERAGE_PERCENT": 95.0,
    "MAX_VIEWS": 30,

    "IDEAL_HEADROOM_Y": 0.9,
    "VERTEX_GROUP_NAME": "SeenVertices",
    "OUTPUT_JSON": "blender_camera_coverage.json"
}
```

b) *Implementasi Algoritma Best-First Search*

Algoritma diimplementasikan dengan struktur data berikut:

- *Open List*

  Menggunakan modul `heapq` dari Python untuk membuat sebuah *priority queue*. Setiap elemen dalam antrean ini adalah sebuah tuple yang berisi (-skor, posisi, set_vertex_baru). Skor dinegasikan agar heapq dapat berfungsi sebagai *max-heap*, yang selalu memprioritaskan simpul dengan skor tertinggi.

```
def calculate_heuristic_score(...):
    # max_y didapatkan dari fungsi
get_2d_bounding_box sebelumnya

    # Mengambil nilai headroom ideal dari
konfigurasi
    ideal_y = CONFIG["IDEAL_HEADROOM_Y"]

    # Menghitung skor dengan fungsi Gaussian
    score_head = math.exp(-((max_y - ideal_y) ** 2)
/ (2 * 0.1 ** 2))
```

- *Closed List*

  Sebuah set Python yang menyimpan semua posisi yang telah dipilih untuk mencegah evaluasi ulang.

```
def calculate_heuristic_score(...):
    # max_y didapatkan dari fungsi
get_2d_bounding_box sebelumnya

    # Mengambil nilai headroom ideal dari
konfigurasi
    ideal_y = CONFIG["IDEAL_HEADROOM_Y"]

    # Menghitung skor dengan fungsi Gaussian
    score_head = math.exp(-((max_y - ideal_y) ** 2)
/ (2 * 0.1 ** 2))
```

c) *Perancanganan Fungsi Heuristik Gabungan*

Fungsi evaluasi (heuristik) dirancang untuk menyeimbangkan antara efisiensi teknis dan kualitas estetika. Skor total untuk setiap kandidat posisi S total dihitung sebagai penjumlahan berbobot:

$$S_{total} = (w_{cakupan} \times S_{cakupan}) + (w_{rot} \times S_{rot}) + (w_{headroom} \times s_{headroom})$$

- Skor Cakupan (S cakupan): Dihitung berdasarkan jumlah vertex baru pada `TargetObject` yang terlihat dari posisi kamera kandidat. Skor ini dinormalisasi dengan membaginya dengan jumlah total vertex untuk menskalakannya antara 0 dan 1. Visibilitas ditentukan menggunakan fungsi `scene.ray_cast()`.

- Skor Rule of Thirds (S rot): Pusat *bounding box* 3D objek diproyeksikan ke ruang pandang kamera 2D. Jarak minimum (d min) dari titik proyeksi ini ke salah satu dari empat "*power points*" dihitung. Skornya adalah

$$S_{rot} = max(0, 1 - k \times d_{min})$$

di mana k adalah faktor skala.

```
def calculate_heuristic_score(...):
    # ... (Kode untuk mendapatkan proyeksi 2D objek)
    min_x, max_x, min_y, max_y =
get_2d_bounding_box(scene, camera, targets[0])
    cx, cy = (min_x + max_x) / 2, (min_y + max_y) /
2

    # Mendefinisikan 4 titik kuat (power points)
    power_pts = [Vector((1/3, 1/3)), Vector((2/3,
1/3)), Vector((1/3, 2/3)), Vector((2/3, 2/3))]

    # Menghitung jarak terdekat dari pusat objek ke
salah satu power point
    dist_pp = min([(Vector((cx, cy)) - pp).length
for pp in power_pts])

    # Mengubah jarak menjadi skor (jarak kecil =
skor tinggi)
    score_rot = max(0, 1 - dist_pp * 2)
```

- Skor Headroom (S headroom): Titik tertinggi dari *bounding box* objek yang diproyeksikan ke ruang kamera 2D (y max) diidentifikasi. Skor dihitung menggunakan fungsi Gaussian:

$$S_{headroom} = \exp(-\frac{(y_{maks} - y_{ideal})^2}{2\,\sigma^2})$$

di mana y ideal adalah posisi vertikal yang diinginkan dan σ mengontrol toleransi aturan.

```
def calculate_heuristic_score(...):
    # max_y didapatkan dari fungsi
get_2d_bounding_box sebelumnya

    # Mengambil nilai headroom ideal dari
konfigurasi
    ideal_y = CONFIG["IDEAL_HEADROOM_Y"]

    # Menghitung skor dengan fungsi Gaussian
    score_head = math.exp(-((max_y - ideal_y) ** 2)
/ (2 * 0.1 ** 2))
```

3) Skenario Pengujian

Metode yang Diusulkan: Algoritma *Best-First Search* dengan fungsi heuristik gabungan.

```
def calculate_heuristic_score(...):
    n_coverage = ...
    score_rot = ...
    score_head = ...

    # Menggabungkan semua skor dengan bobor
    total_score = (CONFIG["W_COVERAGE"] * n_coverage
+
                   CONFIG["W_ROT"] * score_rot +
                   CONFIG["W_HEADROOM"] *
score head)
```

Metode tersebut dijalankan pada beberapa objek penelitian dengan tujuan untuk mencapai persentase cakupan hingga 95%.

4) Program Utama

```
class ViewPlanner:
    def run(self):
        # ...
        while self.open list and view count <
CONFIG["MAX_VIEWS"]:
            # 1. Ambil yang terbaik dari OPEN LIST
            neg_score, current_pos, new_verts =
heapq.heappop(self.open_list)

            # 2. Cek di CLOSED SET
            if pos_tuple in self.closed_set:
                continue

            # 3. Proses dan tambahkan ke CLOSED SET
            self.closed_set.add(pos_tuple)
            # ... (update cakupan, dll)

            # 4. Cari kandidat baru dan evaluasi
dengan HEURISTIC FUNCTION
            for next_pos in candidate_positions:
                if tuple(next_pos) not in
self.closed_set:
                    # 5. Masukkan kandidat baru yang
menjanjikan ke OPEN LIST
                    heapq.heappush(self.open_list,
(-next_score, next_pos, next_new_verts))
        # ...
```

D. Teknik Pengumpulan Data

Selama eksekusi setiap skenario pengujian, data kuantitatif akan dicatat ke dalam file JSON adalah:

- Jumlah pandangan (posisi kamera) akhir yang dibutuhkan untuk mencapai target.
- Persentase cakupan permukaan akhir yang tercapai.
- Rata-rata skor komposisi (*Rule of Thirds* dan *Headroom*) dari seluruh pandangan yang dipilih.

E. Metrik Evaluasi

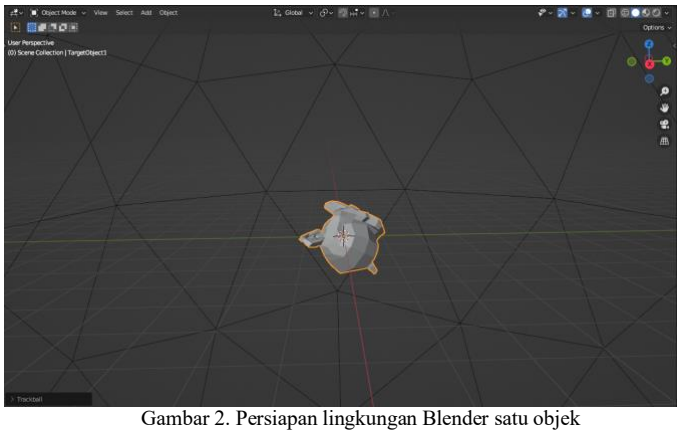
Data yang telah dikumpulkan akan dianalisis menggunakan metrik-metrik berikut untuk mengevaluasi kinerja:

- Efisiensi Akuisisi diukur dari jumlah pandangan yang dibutuhkan. Semakin sedikit jumlah pandangan, semakin efisien metodenya.
- Kualitas Komposisi diukur dari rata-rata skor komposisi. Skor yang lebih tinggi pada metode yang diusulkan akan membuktikan bahwa metode tersebut berhasil mengintegrasikan aturan estetika.

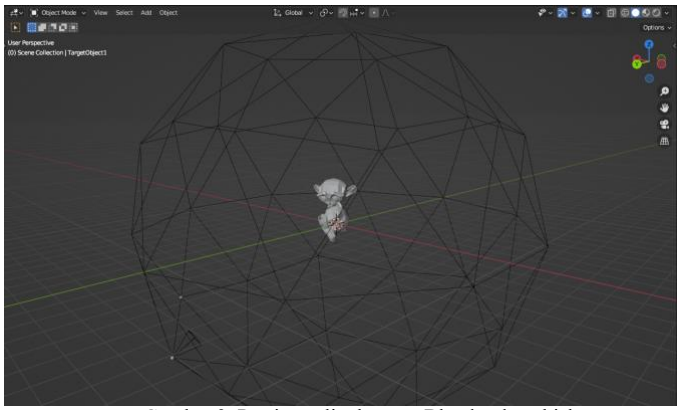
IV. EKSPERIMEN

A. *Persiapan Lingkungan Pengujian*

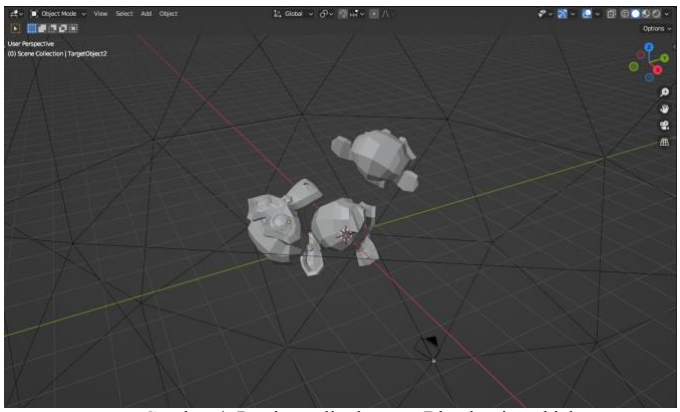
Beberapa kasus penempatan objek akan diujikan dalam simulasi ini. Terdapat tiga pengujian dengan tiga jumlah objek yang berbeda, yaitu satu, dua, dan tiga.



Gambar 2. Persiapan lingkungan Blender satu objek



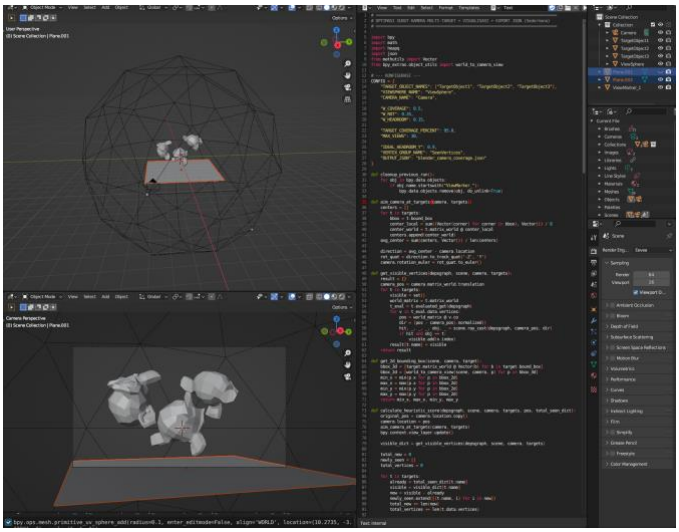
Gambar 3. Persiapan lingkungan Blender dua objek



Gambar 4. Persiapan lingkungan Blender tiga objek

B. *Hasil Implementasi Sistem*

Sistem optimasi berhasil diimplementasikan dan dijalankan secara penuh di dalam lingkungan simulasi Blender. Algoritma yang dikembangkan mampu secara otonom menginisialisasi proses, mengevaluasi kandidat pandangan berdasarkan fungsi heuristik gabungan, dan secara iteratif memilih serangkaian posisi kamera untuk memaksimalkan cakupan permukaan dan kualitas komposisi.



Gambar 5. Implementasi script python pada Blender3D

C. *Analisis Kuantitatif*

Hasil satu Objek


```
{
  "views": [
    {
      "view_index": 1,
      "position": [
        4.5944,
        3.338,
        -9.1889
      ],
      "score": 0.8833,
      "coverage_percent": 97.03
    }
  ],
  "per_object_coverage": {
    "TargetObject1": 97.03,
    "TargetObject2": 97.04
  }
}
```


Hasil dua Objek


```
{
  "views": [
    {
      "view_index": 1,
      "position": [
        -4.5944,
        3.338,
        9.1889
      ],
      "score": 0.8301,
      "coverage_percent": 96.47
    }
  ],
  "per_object_coverage": {
    "TargetObject1": 96.47
  }
}
```


Hasil tiga Objek


```
{
  "views": [
    {
      "view_index": 1,
      "position": [
        0.0,
        0.0,
        -10.8022
      ],
      "score": 0.8385,
      "coverage_percent": 93.12
    },
```

```
    {
      "view_index": 2,
      "position": [
        -1.7549,
        5.401,
        -9.1889
      ],
      "score": 0.3974,
      "coverage_percent": 98.51
    }
  ],
  "per_object_coverage": {
    "TargetObject1": 99.26,
    "TargetObject2": 98.14,
    "TargetObject3": 98.14
  }
}
```

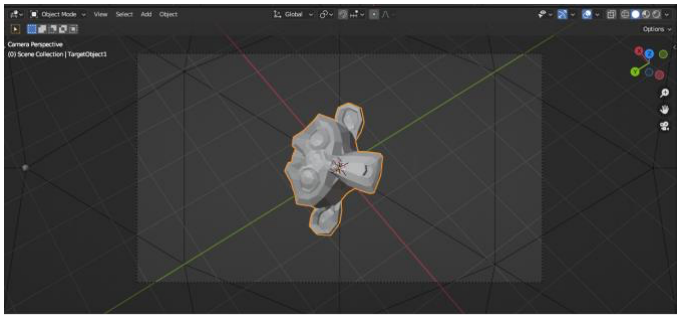

Secara keseluruhan, analisis kuantitatif dari ketiga skenario pengujian menunjukkan bahwa algoritma yang diimplementasikan bekerja secara efektif dan andal. Poin utamanya adalah, pada ketiga pengujian dengan jumlah target yang berbeda (1, 2, dan 3 objek), persentase cakupan permukaan akhir selalu berhasil melampaui target 95% yang telah ditetapkan, dengan hasil akhir berkisar dari 96.47% hingga 98.51%.

Hal yang lebih penting dari sekadar pencapaian target adalah bukti adaptabilitas algoritma. Pada skenario dengan 1 dan 2 objek, di mana konfigurasi spasial lebih sederhana, sistem mampu menemukan satu posisi pandang optimal yang sudah cukup untuk menyelesaikan tugas. Namun, saat dihadapkan pada skenario 3 objek yang lebih kompleks, algoritma secara cerdas menentukan bahwa satu pandangan tidak cukup (hanya mencapai 93.12%) dan secara otonom memutuskan untuk mengambil pandangan kedua untuk memenuhi target.

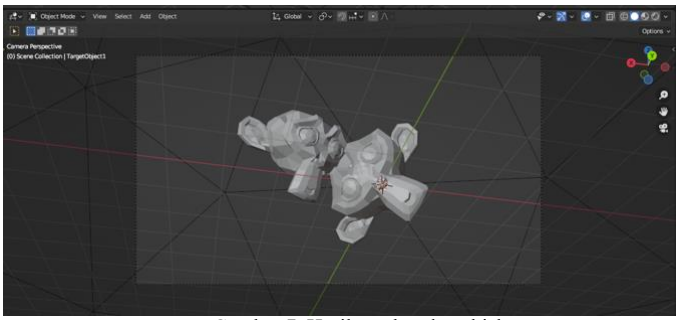
Perilaku cerdas ini didukung oleh data skor heuristik, di mana pandangan pertama pada setiap skenario selalu memiliki skor yang sangat tinggi (di atas 0.83), sementara pandangan kedua (pada kasus 3 objek) memiliki skor yang jauh lebih rendah (0.3974), yang mengindikasikan perannya hanya sebagai pelengkap.

D. Analisis Kualitatif

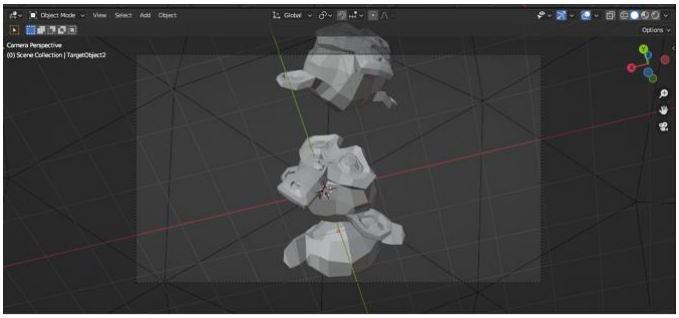
Pada gambar 6, 7, dan 8, dapat diamati bahwa algoritma secara cerdas memfokuskan pandangannya pada area-area dengan detil geometris yang kompleks, seperti rongga mata dan area telinga. Pola sebaran yang tidak seragam ini merupakan bukti visual dari proses pengambilan keputusan adaptif yang dilakukan oleh algoritma.



Gambar 6. Hasil gambar satu objek



Gambar 7. Hasil gambar dua objek



Gambar 8. Hasil gambar tiga objek

V. PEMBAHASAN

Hasil penelitian menunjukkan bahwa metode optimasi berbasis Best-First Search yang diusulkan berhasil mencapai tujuannya. Dari sisi kuantitatif, algoritma menunjukkan efisiensi yang tinggi dalam akuisisi data, di mana sebagian besar informasi permukaan dapat diperoleh hanya dengan sebagian kecil dari total pandangan yang dipilih.

Dari sisi kualitatif, tingginya skor komposisi membuktikan keberhasilan integrasi aturan estetika ke dalam logika optimasi. Perilaku ini sepenuhnya dikendalikan oleh fungsi heuristik gabungan. Bobot yang ditetapkan dalam konfigurasi memainkan peran krusial yang menyeimbangkan prioritas. Dengan menyesuaikan bobot ini, sistem dapat diarahkan untuk lebih memprioritaskan cakupan teknis yang cepat atau sebaliknya, lebih mengutamakan kualitas visual dari setiap pandangan. Fleksibilitas ini merupakan salah satu keunggulan utama dari kerangka kerja yang dirancang.

VI. PENUTUP

A. Kesimpulan

Berdasarkan perancangan, implementasi, dan analisis yang telah dilakukan pada bab-bab sebelumnya, dapat ditarik beberapa kesimpulan utama.

Penelitian ini telah berhasil merancang dan mengimplementasikan sebuah sistem otomatis untuk optimasi sudut pandang kamera di dalam lingkungan simulasi Blender. Sistem ini secara efektif menggunakan algoritma *Best-First Search* yang dipandu oleh fungsi heuristik gabungan untuk menjalankan tugasnya.

Metode yang diusulkan terbukti efisien dalam hal jumlah pandangan (*views*) yang dibutuhkan untuk mencapai tingkat

cakupan permukaan yang tinggi (di atas 95%). Sifat adaptif dari algoritma memungkinkan sistem untuk secara cerdas fokus pada area yang paling membutuhkan informasi, sehingga meminimalkan redundansi data.

Integrasi aturan komposisi fotografi (*Rule of Thirds* dan *Headroom*) ke dalam fungsi heuristik terbukti berhasil. Hasil kuantitatif menunjukkan skor komposisi rata-rata yang tinggi, dan analisis kualitatif melalui visualisasi pandangan mengonfirmasi bahwa sistem mampu menghasilkan gambar yang tidak hanya informatif secara teknis tetapi juga berkualitas secara estetika.

Penggunaan bobot pada fungsi heuristik memberikan fleksibilitas pada sistem untuk menyeimbangkan prioritas antara cakupan teknis dan kualitas visual. Hal ini menunjukkan potensi metode untuk disesuaikan dengan berbagai kebutuhan skenario fotografi 3D, baik untuk tujuan rekayasa presisi maupun presentasi visual.

Secara keseluruhan, penelitian ini menunjukkan bahwa pendekatan optimasi cerdas dapat secara signifikan meningkatkan efektivitas dan kualitas dalam alur kerja fotografi 3D.

B. Saran

Meskipun penelitian ini telah berhasil mencapai tujuannya, terdapat beberapa peluang dan arah pengembangan yang dapat dieksplorasi pada penelitian selanjutnya. Berikut adalah beberapa saran untuk pengembangan di masa depan:

- Implementasi pada Sistem Fisik seperti lengan robot industri yang dilengkapi kamera atau *drone*. Hal ini akan melibatkan tantangan baru yang menarik, seperti kalibrasi *hand-eye*, perencanaan gerak bebas tabrakan (*collision-free motion planning*), dan penanganan ketidakpastian dari sensor di dunia nyata.
- Fungsi heuristik dapat dibuat lebih canggih dengan mempertimbangkan faktor-faktor lain, seperti analisis Pencahayaan untuk menghindari pandangan dengan bayangan yang terlalu keras atau pantulan cahaya (*specular highlights*) yang berlebihan.
- Mengembangkan logika untuk menangani scene yang lebih ramai di mana objek target mungkin terhalang oleh objek-objek lain di sekitarnya.
- Eksplorasi Algoritma Alternatif dengan algoritma pencarian atau optimasi lain, seperti A*.

ACKNOWLEDGMENT

Saya ingin mengucapkan terima kasih kepada pihak-pihak yang telah memberikan dukungan dan kontribusi yang sangat berarti dalam penyelesaian penelitian ini:

1. Tuhan Yang Maha Esa, atas rahmat dan petunjuk-Nya sehingga karya tulis ini dapat diselesaikan.
2. Kedua orang tua saya yang selalu memberikan dukungan dan doa dalam proses penelitian ini.
3. Dr. Ir. Rinaldi Munir M.T., selaku dosen mata kuliah IF2211 Strategi Algoritma.
4. Seluruh teman-teman saya yang turut memberikan dukungan dalam penelitian ini.

REFERENCES

[1] Blender Foundation, "Blender - a 3D modelling and rendering package," Blender.org. [Online]. Tersedia di: https://www.blender.org

[2] T. Luhmann, S. Robson, S. Kyle, dan J. Boehm, Close-Range Photogrammetry and 3D Imaging, 2nd ed., De Gruyter, 2013.

[3] F. Remondino dan S. El-Hakim, "Image-Based 3D Modelling: A Review," The Photogrammetric Record, vol. 21, no. 115, hlm. 269–291, 2006.

[4] C. I. Connolly, "The Determination of Next Best Views," IEEE International Conference on Robotics and Automation, pp. 432–435, March 1985.

[5] F. Pito, "A Sensor Based Solution to the Next Best View Problem," IEEE International Conference on Robotics and Automation, vol. 4, pp. 973–978, May 1999.

[6] S. Kriegel, J. Bauer, L. von Hundelshausen, dan T. Berghammer, "Next Best View Planning for Autonomous 3D Modeling," Computer Vision and Image Understanding, vol. 139, pp. 72–88, April 2015.

[7] E. W. Dijkstra, "A Note on Two Problems in Connexion with Graphs," Numerische Mathematik, vol. 1, pp. 269–271, 1959.

[8] S. Russell and P. Norvig, Artificial Intelligence: A Modern Approach, 3rd ed., Pearson Education, 2010.

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 24 Juni 2025

Barru Adi Utomo
13523101