BAB 2
TINJAUAN PUSTAKA

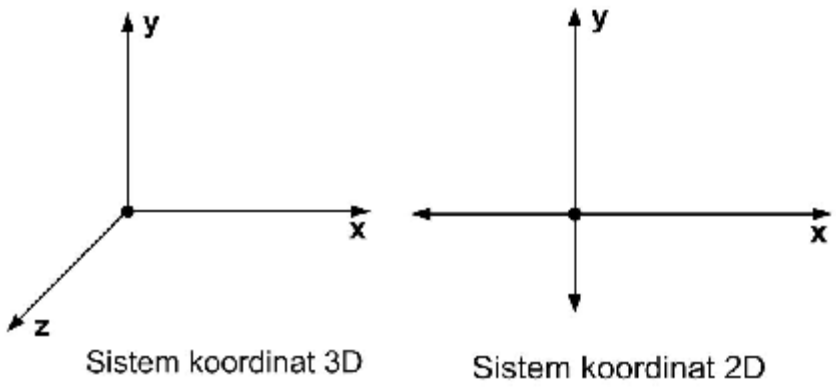
2.1 Landasan Teori

2.1.1 Grafika Komputer

Menurut Janner, grafika komputer pada dasarnya adalah suatu bidang komputer yang mempelajari cara-cara untuk meningkatkan dan memudahkan komunikasi antara manusia dan mesin (komputer) dengan jalan membangkitkan, menyimpan, dan memanipulasi gambar model suatu objek menggunakan komputer. [14]

2.1.1.1 Sistem Koordinat

Dalam dunia nyata, kita mengenal adanya sistem koordinat kartesian yang digunakan untuk membedakan lokasi atau posisi sembarang titik atau objek yang lain. Dalam sistem tersebut dikenal sistem koordinat dua dimensi dan tiga dimensi. Dalam sistem koordinat kartesian dua dimensi, lokasi setiap titik ditentukan oleh besaran. Jika digambarkan, nilai dua besaran tersebut akan membentuk suatu sumbu koordinat mendatar (sumbu X) dan tegak (sumbu Y). Sedangkan, dalam sistem koordinat tiga dimensi selain terdapat sumbu X dan sumbu Y terdapat satu sumbu lain, yang arahnya tegak lurus dengan sumbu X dan sekaligus tegak lurus dengan sumbu Y sehingga lebih tepat bila dikatakan sumbu tersebut tegak lurus (menembus) bidang datar dan disebut sumbu Z.



Gambar 2. 1 Sistem Koordinat Kartesian [14]

2.1.1.2 2 Dimensi

Gambar 2 dimensi memiliki karakteristik hanya memperlihatkan sumbu X dan sumbu Y, atau sumbu koordinat kartesius dua dimensi. Karakteristik kedua adalah bahwa gambar yang dihasilkan seolah "mati". Artinya, tidak dilakukan manipulasi atau pengoperasian tertentu, baik pada objek maupun pada sistem koordinatnya. [14]

2.1.1.3 3 Dimensi

3 dimensi berbeda dengan 2 dimensi yang memperlihatkan sumbu X dan sumbu Y Obyek. 3 dimensi merupakan sekumpulan titik-titik 3-D (x,y,z) yang membentuk luasan-luasan yang digabungkan menjadi satu kesatuan. Terapan-terapan 3 dimensi berbeda dengan terapan 2 dimensi, tidak hanya karena penambahan dimensi dari dua menjadi tiga, tetapi lebih utama adalah cara menampilkan suatu relita dari objek yang sebenarnya ke layar tampilan. [14]

Macam-macam transformasi 3 dimensi sebagai berikut:

1. Translasi

   Menurut Basuki, translasi adalah proses untuk memindahkan obyek ke arah sumbu X, sumbu Y dan sumbu Z sebesar (dx,dy,dz). [3]

2. Rotasi

   Pemutaran atau rotasi 3 dimensi lebih rumit dibandingkan pemutaran 2 dimensi, tetapi pada dasarnya sama yaitu bahwa pemutaran bisa dilaksanakan dengan memilih salah satu sumbu koordinasi sebagai sumbu putar. [14]

3. Penskalaan

   Proses pembesaran atau perkecil gambar. Penskalaan bisa dilaksanakan ke arah x saja, ke arah y saja, ke arah z saja, atau kombinasi ketiga-tiganya.

2.1.1.4 Pemodelan 3D

Pemodelan adalah membentuk suatu benda-benda atau obyek. Membuat dan mendesain obyek tersebut sehingga terlihat seperti hidup. Sesuai dengan obyek dan basisnya, proses ini secara keseluruhan dikerjakan di komputer.

Melalui konsep dan proses desain, keseluruhan obyek bisa diperlihatkan secara 3 dimensi, sehingga banyak yang menyebut hasil ini sebagai pemodelan 3 dimensi (3D modelling).

Ada beberapa aspek yang harus dipertimbangkan bila membangun model obyek, kesemuanya memberi kontribusi pada kualitas hasil akhir. Hal-hal tersebut meliputi metode untuk mendapatkan atau membuat data yang mendeskripsikan obyek, tujuan dari model, tingkat kerumitan, perhitungan biaya, kesesuaian dan kenyamanan, serta kemudahan manipulasi model.

Proses pemodelan 3D membutuhkan perancangan yang dibagi dengan beberapa tahapan untuk pembentukannya, antara lain : [12]

a. Obyek apa yang ingin dibentuk sebagai obyek dasar
b. Metode pemodelan obyek 3D

Teknik *Polygonal modeling* adalah teknik membuat model dengan memakai objek-objek *geometry* dasar yang kemudian dikembangkan menjadi objek model yang lebih kompleks. Umumnya memakai bentuk objek *geometry box* (kotak) yang kemudian dihaluskan lagi permukaannya (*smooth*).

Teknik NURBS modeling adalah teknik membuat model dengan memakai garis-garis yang dibuat seperti rangka objek yang diinginkan kemudian diberi surface (bentuk permukaan).

c. Pencahayaan dan animasi gerakan obyek sesuai dengan urutan proses yang akan dilakukan. Ada 3 teknik pencahayaan pada 3 dimensi yaitu:
    1. Lampu utama (*Key light*)

       Key Light merupakan pencahayaan utama dari gambar kita, dan merepresentasikan bagian paling terang sekaligus mendefiniskan bayangan pada gambar. *Key Light* juga merepresentasikan pencahayaan paling dominan seperti matahari dan lampu interior. Meski demikian peletakannya tidak harus persis tepat pada sumber pencahayaan yang kita inginkan.

2. Lampu pengisi (*Fill Light*)

   Fungsi *fill light* adalah melembutkan sekaligus mengisi bagian gelap yang diciptakan oleh *key light*. *Fill Light* juga berfungsi menciptakan kesan tiga dimensi.

3. Cahaya latar (*Back Light*)

   Back Light berfungsi untuk menciptakan pemisahan antara objek utama dengan objek pendukung. Dengan diletakkan pada bagian belakang benda back light menciptakan "garis pemisah" antara objek utama dengan latar belakang pendukungnya.

2.1.2 *Augmented Reality* (AR)

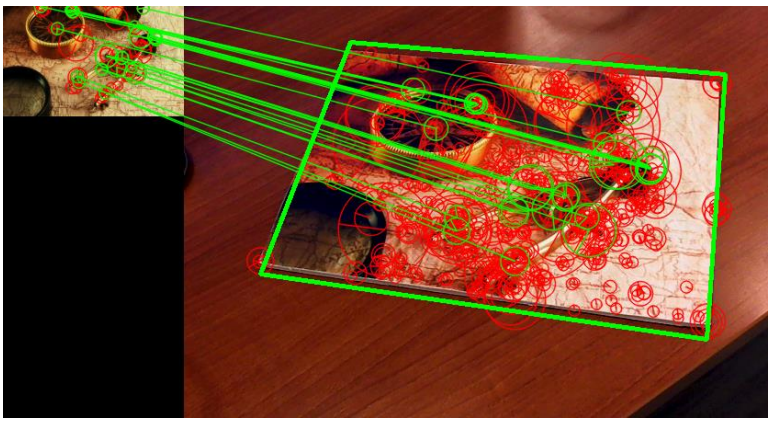
Ronald T. Azuma mendefinisikan *Augmented Reality* sebagai penggabungan benda-benda nyata dan maya dilingkungan nyata, berjalan secara interaktif dalam waktu nyata, dan terdapat integrasi antar benda dalam tiga dimensi, yaitu benda maya terintegrasi dalam dunia nyata. Tidak seperti realitas maya yang sepenuhnya menggantikan kenyataan, namun *augmented reality* hanya menambahkan atau melengkapi kenyataan. Ronald T. Azuma juga mendefinisikan *Augmented reality* sebagai sistem yang memiliki karakteristik sebagai berikut : [2]

a. Menggabungkan lingkungan nyata dan virtual
b. Berjalan secara interaktif dalam waktu nyata
c. Integrasi dalam tiga dimensi (3D).

Penggabungan objek nyata dan virtual dimungkinkan dengan teknologi *display* yang sesuai, interaktivitas dimungkinkan melalui perangkat-perangkat *input* tertentu. Jika menelisik lebih dalam, sesungguhnya AR berbeda dengan Virtual Reality. Virtual *reality* adalah teknologi yang memungkinkan seseorang melakukan simulasi terhadap suatu objek nyata dengan menggunakan komputer yang mampu membangkitkan suasana tiga dimensi (3D) sehingga membuat pemakai seolah-olah terlibat secara fisik. AR merupakan variasi dari *Virtual Environments* (VE), atau yang lebih dikenal dengan istilah *Virtual Reality* (VR).

2.1.3 *Markerless*

Markerless sangat berbeda dengan *fiduciary marker*. Menurut Madden [11]. *markerless* adalah AR yang digunakan untuk melacak objek yang ada di dunia nyata tanpa marker yang spesial. Untuk melakukan pelacakan obyek, sistem AR *markerless,* bergantung pada natural feature-tracking. AR *markerless* melakukan pelacakan dengan obyek yang sudah ada sebelumnya yang kemudian dijadikan *image target* atau gambar yang akan dideteksi untuk menampilkan obyek virtual [1].



Gambar 2. 2 Markerless [1]

2.1.4 Vuforia

Vuforia merupakan *library* untuk *Augmented Reality* yang dikembangkan oleh Qualcomm, yang menggunakan sumber yang konsisten mengenai *Computer Vision* yang fokus pada *image recognition.* Vuforia mempunyai banyak fitur-fitur dan kemampuan, yang dapat membantu pengembang untuk mewujudkan pemikiran mereka tanpa adanya batas secara teknikal.

Dengan *support* untuk kios, Android dan Unity3D, *platform* Vuforia mendukung para pengembang untuk membuat aplikasi yang dapat digunakan di hampir seluruh jenis *smartphone* dan *tablet*.

Pengembang juga diberikan kebebasan untuk mendesain dan membuat aplikasi yang mempunyai kemampuan antara lain : [7]

1. Teknologi *Computer Vision* tingkat tinggi yang mengijinkan *developer* untuk membuat efek khusus pada *mobile Service*.

2. Terus-menerus mengenali *multiple image.*
3. *Tracking* dan *Detection* tingkat lanjut.
4. Solusi pengaturan *database* gambar yang fleksibel.

Target pada Vuforia merupakan obyek pada dunia nyata yang dapat dideteksi oleh kamera, untuk menampilkan obyek virtual. Beberapa jenis target pada Vuforia adalah : [7]

1. *Image targets*, contoh : foto, papan permainan, halaman majalah, sampul buku, kemasan produk, poster dan kartu ucapan.
2. *Frame markers*, contoh : tipe *Frame* gambar 2D dengan *pattern* khusus yang dapat digunakan sebagai permainan.
3. *Multi-target*, contoh : kemasan produk atau produk berbentuk kotak ataupun persegi. Jenis ini dapat menampilkan gambar sederhana *Augmented* 3D.
4. *Virtual buttons*, yang dapat membuat tombol sebagai daerah kotak sebagai sasaran gambar.

Pada vuforia, ada 2 jenis *workflow* dengan dasar *database* yang dapat dipilih oleh *developer*, yaitu *Cloud Database* dan *Device Database*.

2.1.4.1 Vuforia API Reference

API reference berisi informasi tentang hierarki kelas dan fungsi ember dari QCAR SDK. Sistem dari QCAR SDK menyediakan callback event. Contoh : sebuah image baru yang tersedia. [7]

a. High-level access ke perangkat keras. Contoh : kamera start / stop.
b. Multiple trackables.
c. Interaksi secara langsung dengan dunia nyata.

2.1.4.2 Arsitektur Vuforia

Vuforia SDK memerlukan beberapa komponen penting agar dapat bekerja dengan baik. Komponen-komponen tersebut antara lain [7] :

1. *Camera*

   Kamera dibutuhkan untuk memastikan bahwa setiap *Frame* ditangkap dan diteruskan secara efisien ke *tracker*. Para *developer* hanya tinggal memberi tahu kamera kapan mereka mulai menangkap dan berhenti.

2. *Image Converter*

   Mengkonversi format kamera ke dalam format yang dapat dideteksi oleh OpenGL dan untuk *Tracking.*

3. *Tracker*

   Mengandung algoritma *Computer Vision* yang dapat mendeteksi dan melacak objek dunia nyata yang ada pada *video* kamera. Berdasarkan gambar dari kamera, algoritma yang berbeda bertugas untuk mendeteksi trackable baru, dan mengevaluasi *virtual button.* Hasilnya akan disimpan dalam *State object* yang akan digunakan oleh *video background renderer* dan dapat diakses dari *application code*.

4. *Video background renderer*

   Ma-render gambar dari kamera yang tersimpan di dalam State object. Performa dari video background renderer sangat bergantung pada *Service* yang digunakan.

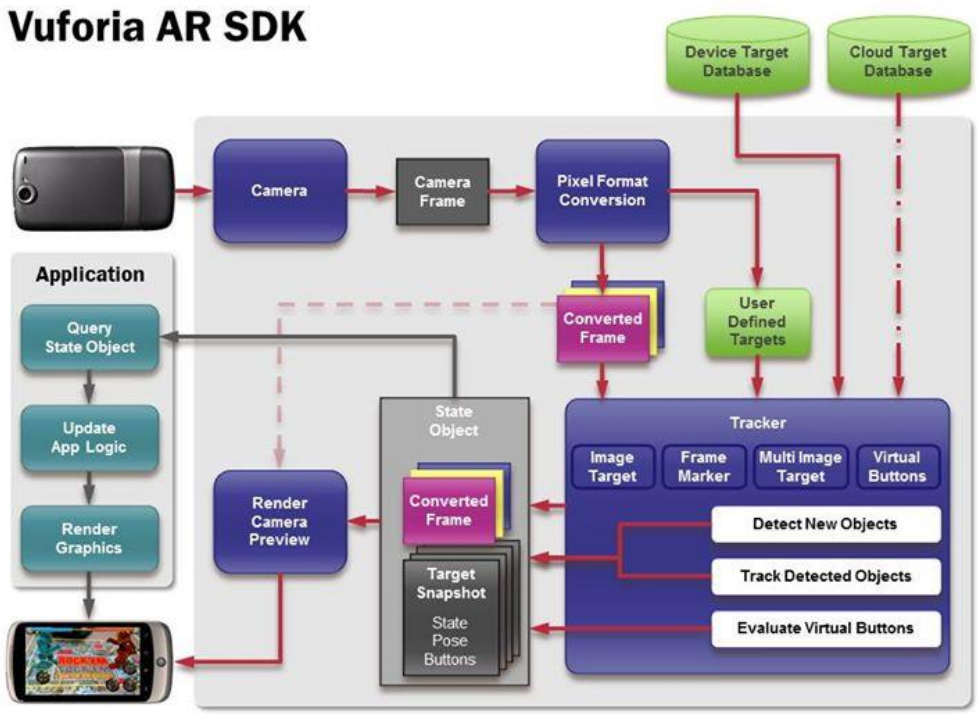
5. *Application code*

   Menginisialisasi semua komponen di atas dan melakukan tiga tahapan penting dalam application code seperti :

   a. Query State object pada target baru yang terdeteksi atau marker.
   b. Update logika aplikasi setiap input baru dimasukkan.
   c. Render grafis yang ditambahkan (*augmented).*

6. *Target Resources*

   Dibuat menggunakan *on-line Target Management System*. *Asset* yang diunduh berisi sebuah konfigurasi xml config.xml yang memungkinkan *developer* untuk mengkonfigurasi beberapa fitur dalam *trackable* dan *binary file* yang berisi *database trackable*.



Gambar 2. 3 Arsitekur Vuforia [7]

2.1.5 *Natural Feature Tracking and Detection*

Dalam metode ini informasi yang diperlukan untuk tujuan pelacakan dapat diperoleh dengan cara *optical-flow* berbasis pencocokan *template* atau korespondensi fitur. "*Optical flow* atau aliran optik adalah pola gerakan jelas benda, permukaan, dan tepi dalam adegan visual yang disebabkan oleh gerakan relatif antara pengamat (mata atau kamera) dan adegan". Korespondensi fitur bekerja lebih baik dan lebih efektif daripada pencocokan template karena mereka bergantung pada pencocokan fitur lokal. Mengingat korespondensi tersebut, pose secara kasar dapat dihitung dengan estimasi yang kuat yang membuatnya cukup sensitif terhadap oklusi parsial, blur, refleksi, perubahan skala, kemiringan, perubahan iluminasi atau kesalahan pencocokan. Salah satu unsur diterapkan pendekatan pelacakan fitur alami didasarkan pada versi modifikasi dari SIFT. SIFT sangat baik dalam mengekstrak tetapi prosesor intensif bekerja karena komputasi.

2.1.5.1 SIFT (*Scale Invariant Feature Transform*)

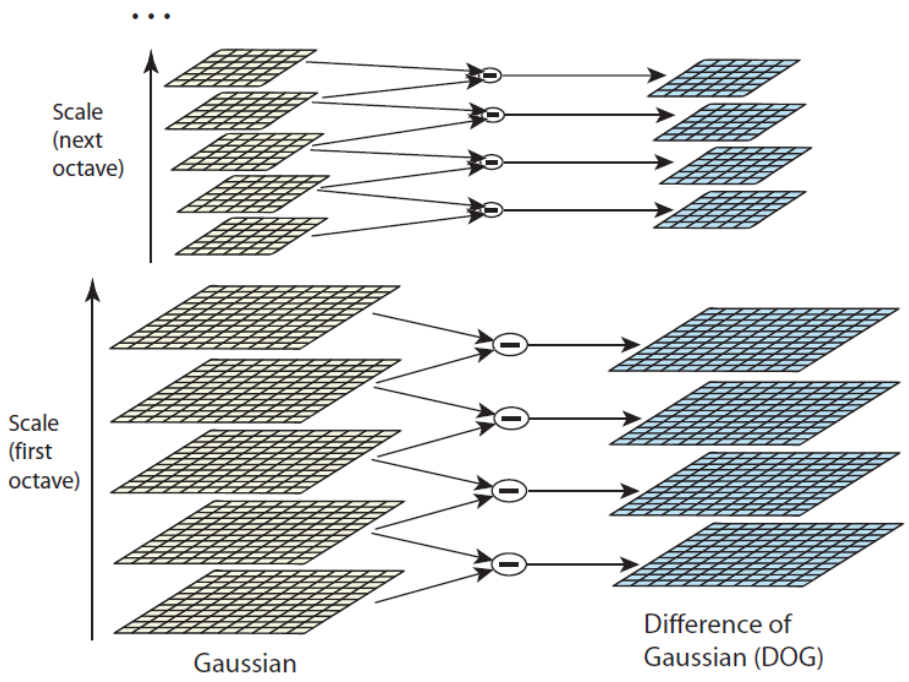
Scale Invariant Feature Transform merupakan metode yang mengambil titik-titik fitur yang ada pada suatu citra, untuk membantu memastikan dalam pencocokan titik fitur suatu obyek pada sudut pandang yang berbeda. Pendekatan ini mengubah sebuah gambar menjadi sebuah koleksi besar dari vektor fitur lokal, yang masing-masing adalah invarian terhadap translasi, scaling, dan rotasi citra, dan sebagian invarian perubahan pencahayaan dan proyeksi 3D [5].

Pendekatan umum yang dilakukan pada algoritma SIFT terdiri dari keypoint lokalisasi, deskripsi fitur dan pencocokan fitur. Dimana algoritma SIFT sendiri terbagi lagi atas beberapa proses yaitu [5] :

1. Pembentukan *Gaussian Scale Space* dan *Difference of Gaussian Scale Space*.
2. Deteksi ekstremum (maksimum atau minimum) pada *Difference of Gaussian Scale Space*.
3. Penetapan orientasi *keypoint* (*orientation assignment*).
4. Penetapan *keypoint descriptor.*

A. Gaussian dan Difference of Gaussian Scale Space

Pada algoritma SIFT digunakan fungsi *Gaussian* untuk membangun *scale space* sehingga disebut dengan *Gaussian Scale Space*. Gambar 3.1 Ilustrasi *Gaussian* dan *Difference of Gaussian Scale Space.*



Gambar 2. 4 Ilustrasi Gaussian dan Difference of Gaussian Scale Space [5]

Mencari nilai Gaussian :

$$G(x,y,k\sigma) = \frac{1}{2\pi(k\sigma)^2} e^{-(x^2+y^2)/2k^2\sigma^2} \quad \text{.....(2.1)}$$

Dimana,

- $k = \sqrt{2}$
- $\sigma = 1{,}6$

Setelah mendapatkan *Gaussian Scale Space* langkah selanjutnya yaitu membentuk *Difference of Gaussian Scale (DoG)*. *DoG* dibentuk dengan cara :

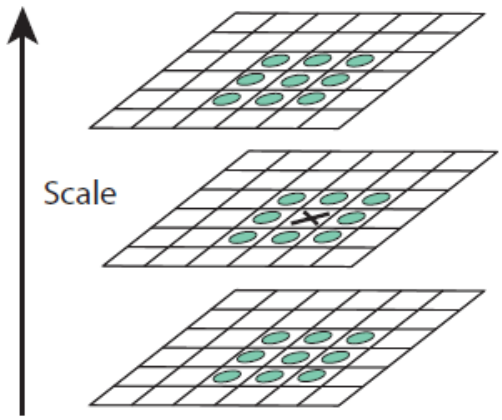
$$D(x,y,\sigma) = L\,(x,y,k\sigma) - L\,(x,y,\sigma) \quad \text{.....(2.2)}$$

Keterangan :

- $L\,(x,y,\sigma)$ = $G\,(x,y,\sigma)$ pada skala σ.
- $L\,(x,y,k\sigma)$ = $G\,(x,y,k\sigma)$ pada skala $k\sigma$ dengan nilai $k = \sqrt{2}$.

B. Deteksi Ekstremum

Deteksi ekstremum (nilai maksimum dan minimum) dilakukan dengan cara membandingkan nilai setiap piksel pada *DoG Scale space* dengan delapan piksel yang berada di sekelilingnya dan 9 piksel yang bersesuaian pada citra DoG sebelum dan setelahnya [5]. Jika nilai piksel yang dimaksud lebih besar atau lebih kecil daripada nilai-nilai piksel pembandingnya maka koordinat piksel tersebut ditandai sebagai ekstremum. Ilustrasinya dapat dilihat pada gambar di bawah ini



Gambar 2. 5 Ilustrasi deteksi ekstrema, tanda silang merupakan piksel uji dan lingkaran hijau merupakan piksel pembanding [5]

Setelah mendapatkan titik-titik ekstremum maka perlu ditingkatkan lokalisasinya dengan akurasi subpiksel dengan menggunakan ekspansi Taylor orde kedua dari fungsi ruang skala , sehingga posisi ekstremum sebenarnya didapatkan dengan rumus berikut [5] :

$$z = -\left(\frac{\partial^2 D}{\partial x^2}\right)^{-1} \frac{\partial D}{\partial x} \quad \text{.....(2.3)}$$

Selanjutnya untuk menghitung nilai *keypoint* pada ekstremum digunakan rumus berikut :

$$D(z) = D + \frac{1}{2}\frac{\partial D^{-1}}{\partial x}\, z \quad \text{.....(2.4)}$$

Jika nilai *D(z)* tidak melebihi suatu nilai *threshold* (0.03) maka *keypoint* tersebut dihilangkan dan tidak dipakai lagi [5]. Di samping penghapusan *keypoint* yang tidak memenuhi syarat *threshold* dilakukan juga penghapusan *keypoint* tak stabil yang berada pada daerah *edge*. Sebuah matriks Hessian 2x2 digunakan untuk menemukan kurvatur sehingga rasio dari kurvatur dapat diketahui [4].

C. Penetapan Orientasi *Keypoint (Orientation assignment)*

Pada penetapan orientasi ini digunakan citra Gaussian *smooth L* yang memiliki skala paling dekat dengan skala *keypoint*. Untuk setiap citra sampel *L(x,y) magnitude m(x,y)* dan orientasi *θ(x,y)* dihitung dengan menggunakan perbedaan piksel sebagai berikut :

$$m(x,y) = \sqrt{\big(L(x+1,y) - L(x-1,y)\big)^2 + \big(L(x,y+1) - L(x,y-1)\big)^2} \quad \text{.....(2.5)}$$

$$\theta(x,y) = \arctan\left(\frac{L(x,y+1) - L(x,y+1)}{L(x+1,y) - L(x-1,y)}\right) \quad \text{.....(2.6)}$$

D. Penentuan Deskriptor Lokal

Proses terakhir adalah menghitung vektor deskriptor. Deskriptor dihitung untuk masing-masing *keypoint*, langkah ini dilakukan pada gambar yang paling dekat dengan skala untuk skala *keypoint*. Pertama membuat orientasi dengan 4x4 piksel dengan 8 bin untuk tiap keypoint [5]. Histogram yang didapat pada langkah penetapan orientasi ini dihitung *magnitude* dan nilai orientasi untuk sampel dalam wilayah 16x16 disekitar *keypoint*. *Magnitude* dihitung dengan fungsi Gaussian dengan σ sama dengan satu setengah lebar deskriptor. Kemudian deskriptor menjadi vektor dari semua nilai histogram ini. Karena 4x4=16 histogram dengan masing-masing memiliki 8bin, maka vektor memiliki 128 elemen.

2.1.5.2 Ferns : *Tracking by Classification*

Ferns digunakan untuk meningkatkan kecepatan dalam mendeteksi "feature" dan melakukan klasifikasi "*marker*/*image target*". Klasifikasi pada pelacakan fitur [6] bekerja dengan belajar mendistribusikan beberapa fitur F dari

sekumpulan *class* C yang sesuai untuk model poin mC dalam model gambar. Pada saat dijalankan, titik bunga yang dideteksi dengan menggunakan pendeteksian titik-titik yang dianggap penting, untuk menghasilkan nilai fitur F yang akan dihitung dan titik yang diklasifikasikan untuk memaksimalkan probabilitas dengan mengamati nilai fitur F.

$$C = argmax\ P\ (Ci|F)\ selama\ Ci$$

Model Titik mC sesuai dengan kelas C dari hasil ini. Titik tersebut kemudian digunakan sebagai korespondensi 3D. Berbeda dengan fitur pendekatan yang lainnya, skema klasifikasi tidak didasarkan pada ukuran jarak , tetapi dilatih untuk mengoptimalkan pendelkatan titik-titik fitur asli model gambar. Skema klasifikasi bisa kurang intensif dalam komputasi, tergantung pada fitur dasar yang digunakan. Ferns [6] merupakan teknik klasifikasi dengan menggunakan biner sebagai fitur yang membandingkan intensitas citra di sekitar poin p yang dianggap penting. Dimana biner dengan fitur F adalah fungsi dari parameter F(p) dengan sepasang pengimbang (l,r) sehingga,

$$F(p) = \begin{cases} 1\ if\ I(p+l) < I(p+r) \\ 0, sebaliknya \end{cases}$$

Untuk nilai N diperoleh dari fitur Fi probabilitas *class* C dapat dihitung dengan menggunakan teorema Bayes.

$$P(C|\{Fi\}) = P(\{Fi\}|C)P(C)/P(\{Fi\}) \quad \text{..... (2.7)}$$

2.1.6 C# (C Sharp)

C# (dibaca: C sharp) merupakan sebuah bahasa pemrograman yang berorientasi objek yang dikembangkan oleh Microsoft sebagai bagian dari inisiatif kerangka .NET *Framework*. Bahasa pemrograman ini dibuat berbasiskan bahasa C++ yang telah dipengaruhi oleh aspek-aspek ataupun fitur bahasa yang terdapat pada bahasa-bahasa pemrograman lainnya seperti Java, Delphi, Visual Basic, dan lain-lain dengan beberapa penyederhanaan. Menurut standar ECMA-334 C# *Language Specification*, nama C# terdiri atas sebuah huruf Latin C (U+0043)

yang diikuti oleh tanda pagar yang menandakan angka # (U+0023). Tanda pagar # yang digunakan memang bukan tanda kres dalam seni musik (U+266F), dan tanda pagar # (U+0023) tersebut digunakan karena karakter kres dalam seni musik tidak terdapat di dalam keyboard standar. [15] Gambar 2.6 merupakan gambar logo C#.



Gambar 2. 6 Logo C# [16]

2.1.7 UML (*Unified Modeling Language*)

UML adalah salah satu bentuk *language* atau bahasa untuk menggambarkan *object oriented analysis and design.* Sebagian besar metode terdiri bahasa dan proses. UML disebut bahasa pemodelan bukan metode. Bahasa pemodelan adalah notasi (terutama grafis) bahwa metode digunakan untuk mengekspresikan desain.



Gambar 2. 7 Logo UML [13]

2.1.7.1 Sejarah UML

Bahasa pemograman berorientasi objek yang pertama dikembangkan dikenal dengan nama Simula-67 yang dikembangkan pada tahun 1967. Bahasa

pemograman ini kurang berkembang dan dikembangkan lebih lanjut, namun kemunculannya menjadi titik besar bagi developer pengembang bahasa pemograman berorientasi objek selanjutnya.

Karena banyaknya metodologi yang berkembang pesat pada saat itu, maka muncullah ide untuk membuat sebuah bahasa yang dapat dimengerti semua orang. Maka dibuatlah bahasa gabungan dari beberapa konsep seperti *Object Modelling Technique (OMT)* dari Rumbaugh dan Booch (1991), konsep *The Classes, Responsibilities, Collaborators (CRC)* dari Rebecca Wirfs-Brock (1990), konsep pemikiran Ivan Jacobson, dan beberapa konsep lainnya dimana James R. Rumbaigh, Grady Booch, dan Ivar Jacobson bergabung dalam sebuah perusahaan yang bernama *Rational Software Corporation* menghasilkan bahasa yang disebut dengan *Unified Modeling Language (UML).* [13]

2.1.7.2 Diagram UML

Menggunakan berbagai macam diagram dengan fungsi masing-masing untuk menggambarkan setiap proses dari sistem berorientasi objek. Berikut merupakan beberapa diagram UML diantaranya : [13]

A. *Use Case Diagram*

Use Case atau diagram *use case* merupakan pemodelan yang digunakan untuk menggambarkan kelakuan (*behavior)* dari sistem yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem yang akan dibuat. Secara kasar, *use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem dan siapa saja yang berhak menggunakan fungsi-fungsi tersebut. Syarat penamaan pada *use case* adalah nama didefinisikan sesimpel mungkin dan dapat dipahami. Ada dua hal utama pada *use case* yaitu pendefinisian apa yang disebut aktor dan *use case*.

1. Aktor merupakan orang, proses, atau sistem lain yang berinteraksi dengan sistem yang akan dibuat diluar sistem yang akan dibuat itu sendiri, jadi walaupun simbol dari actor adalah gambar orang, tapi actor belum tentu merupakan orang.

2. *Use case* merupakan fungsionalitas yang disediakan sistem sebagai unit-unit yang saling vertukar pesan antarunit atau actor.

B. *Activity Diagram*

Diagram aktivitas atau *activity* diagram adalah sebuah diagram yang menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis. Dalam diagram aktivitas yang perlu diperhatikan adalah bahwa diagram aktivitas menggambarkan aktivitas sistem, bukan apa yang dilakukan aktor, jadi aktivitas yang dapat dilakukan oleh sistem.

Diagram aktivitas juga banyak digunakan untuk mendefinisikan hal-hal berikut:

1. Rancangan proses bisnis di mana setiap urutan aktivitas yang digambarkan merupakan proses bisnis system yang didefinisikan
2. Urutan atau pengelompokan tampilan dari sistem/*user interface* di mana setiap aktivitas dianggap memiliki sebuah rancangan antaruka tampilan.
3. Rancangan pengunjian di mana setiap aktivitas dianggap memerlukan sebuah pengujian yang perlu didefinisikan kasus ujinya.

C. *Class Diagram*

Diagram kelas atau *class* diagram menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Kelas memiliki apa yang disebut atribut dan metode atau operasi.

1. Atribut merupakan variabel-variabel yang dimiliki oleh suatu kelas
2. Operasi atau metode adalah fungsi-fungsi yang dimiliki oleh suatu kelas

D. *Sequence Diagram*

Diagram sekuen adalah diagram yang menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *message* yang dikirimkan dan diterima antarobjek. Oleh karena itu untuk menggambarkan diagram sekuen maka harus diketahui objek-objek yang terlibat dalam sebuah *use case* beserta metode-metode yang dimiliki kelas yang d2nstansiasi menjadi objek itu.

Banyaknya diagram sekuen yang harus digambarkan adalah sebanyak pendefinisian *use case* yang memiliki prose situ sendiri atau yang penting semua *use case* yang telah didefinisikan interaksi jalannya pesan sudah dicakup pada diagram sekuen sehingga semakin banyak *use case* yang didefinisikan maka diagram sekuen yang harus dibuat juga semakin banyak.

2.1.8 Software Pendukung

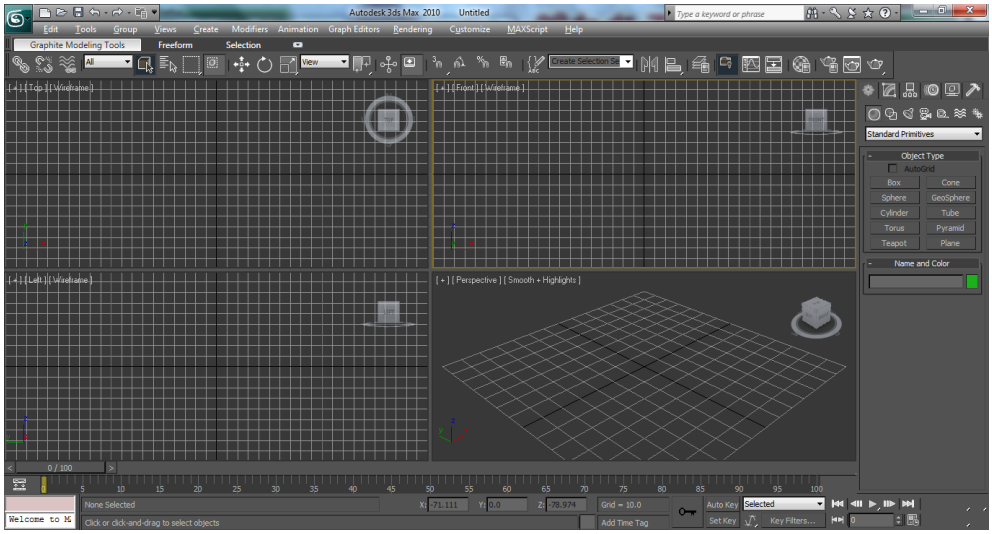
Berikut ini akan dijelaskan mengenai *software-software* pendukung yang digunakan untuk membangun aplikasi yang akan dibuat. *Software* yang digunakan antara lain seperti *Autodesk 3ds Max 2010,* dan *Unity3D.*

2.1.8.1 Autodesk 3ds Max 2010

Autodesk 3ds Max 2010 adalah software pembuatan model animasi tiga dimensi, rendering dan pembuatan grafis pada game dan animasi. Autodesk 3ds Max 2010 banyak digunakan oleh para praktisi dalam bisnis periklanan. Software ini banyak ragamnya, sesuai dengan keterserdiaan fasiltas yang disediakan untuk memudahkan pengguna. Discreet 3DS Max merupakan software tiga dimensi yang dapat membuat objek tiga dimensi tampak realistis. Keunggulan yang dimiliki adalah kemampuannya dalam menggabungkan objek image,vektor dan tiga dimensi serta langsung dapat objek tersebut. Animasi tiga dimensi dapat diintegrasikan pada halaman multimedia dan juga bisa berdiri sendiri sebagai sebuah movie.

Autodesk 3ds Max, 3D Studio MAX sebelumnya, adalah pemodelan, animasi dan rendering paket yang dikembangkan oleh Autodesk Media dan Entertainment. Autodesk memiliki kemampuan pemodelan, arsitektur plugin yang fleksibel dan dapat digunakan pada platform Microsoft Windows. Software Ini sering digunakan oleh pengembang video animation, studio TV komersial dan studio visualisasi arsitektur. Hal ini juga digunakan untuk efek-efek film dan film pra-visualisasi. Selain pemodelan dan tool animasi, versi terbaru dari 3DS Max juga memiliki fitur shader (seperti *ambient occlusion* dan *subsurface scattering*), *dynamic simulation, particle systems, radiosity,* normal *map creation, rendering,*

global illumination, customize user interface, dan bahasanya *scripting* untuk 3Dmax [8].



Gambar 2. 8 Autodesk 3ds Max 2010 [8]

2.1.8.2 Unity3D

Unity adalah sebuah sebuah *tool* yang terintegrasi untuk membuat *game*, arsitektur bangunan dan simulasi. Unity bisa untuk games PC dan *games Online*. Untuk *games Online* memerlukan sebuah *plugin*, yaitu Unity *Web Player*, sama halnya dengan *Flash Player* pada *Browser*. Bahasa pemrograman yang digunakan bermacam-macam, mulai dari Javascript, C#, dan Boo [17]**.**

Pada unity, tidak bisa melakukan desain / *modelling*, dikarenakan unity bukan *tool* untuk mendesain Jadi jika ingin mendesain, harus memerlukan 3D editor lain seperti 3dsmax atau Blender. Banyak hal yang bisa di lakukan di unity , ada fitur audio *reverb zone* , *particle effect* , *Sky Box* Untuk menambahkan langit , dan masih banyak lagi tentunya, juga bisa langsung mengedit *texture* dari *editor* seperti photoshop dll , unity bagus untuk pemula maupun *expert*.

1. Mendukung 3 bahasa pemrograman, JavaScript, C#, dan Boo.
2. *Flexible* and *EasyMoving*, *rotating*, dan *scaling objects* hanya perlu sebaris kode . Begitu juga dengan *Duplicating*, *removing*, dan *changing properties*.

3. *Multi Platform Game* bisa di *deploy* di PC, Mac, Wii, iPhone, iPad dan *browser*, android.
4. *Visual Properties Variables* yang di definisikan dengan *scripts* ditampilkan pada *Editor*. Bisa digeser, di *drag and drop*, bisa memilih warna dengan *color picker*.
5. Berbasis .NET –> Penjalanan program dilakukan dengan *Open Source* .NET platform, Mono.



Gambar 2. 9 Unity3D [17]