



PROPOSAL TESIS

**PENCARIAN JALUR TERPENDEK PADA 3D  
*VISIBILITY ROADMAP* DENGAN PENGHALUSAN  
JALUR BERDASARKAN KEMAMPUAN MANUVER  
*FIXED WING***

DILA MARTA PUTRI  
07111850020001

DOSEN PEMBIMBING  
Dr. Trihastuti Agustinah, S.T., M.T

PROGRAM MAGISTER  
MEDAN KEAHLIAN TEKNIK SISTEM PENGATURAN  
DEPARTEMEN TEKNIK ELEKTRO  
FAKULTAS TEKNOLOGI ELEKTRO  
INSTITUT TEKNOLOGI SEPULUH NOPEMBER  
SURABAYA  
2020



## **LEMBAR PENGESAHAN PROPOSAL TESIS**

**Judul : Pencarian Jalur Terpendek pada 3D Visibility Roadmap dengan  
Penghalusan Jalur Berdasarkan Kemampuan Manuver Fixed Wing**  
**Oleh : Dila Marta Putri**  
**NRP : 07111850020001**

**Telah diseminarkan pada**

**Hari : Rabu**  
**Tanggal : 18 Desember 2019**  
**Tempat : AJ-204**

**Mengetahui/menyetujui**

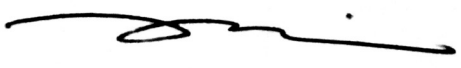
**Dosen Penguji:**

**Calon Dosen Pembimbing**



**1. Prof. Ir. Abdullah Alkaff M.Sc., Ph.D**  
**NIP. 19550123 198003 1 002**

**1. Dr. Trihastuti Agustinah, S.T., M.T**  
**NIP. 19680812 199403 2 001**



**2. Dr. Ir. Ari Santoso, DEA**  
**NIP. 19660218 199102 1 001**

*Halaman ini sengaja dikosongkan*

# **PENCARIAN JALUR TERPENDEK PADA 3D *VISIBILITY ROADMAP* DENGAN PENGHALUSAN JALUR BERDASARKAN KEMAMPUAN MANUVER *FIXED WING***

Nama mahasiswa : Dila Marta Putri  
NRP : 07111850020001  
Pembimbing : 1. Dr. Trihastuti Agustinah, S.T., M.T

## **ABSTRAK**

Penelitian ini memperkenalkan sebuah algoritma Theta\* yang menemukan jalur terpendek untuk sistem *Fixed wing* menuju titik target pada sebuah lingkungan perkotaan yang di buat dalam bentuk 3D pada *visibility roadmap*, lingkungan perkotaan dianggap sebagai hambatan statis untuk menuju titik target, Untuk menghilangkan osilasi dan belokan yang kencang di lintasan yang dihasilkan *Fixed wing*, kami mengusulkan pendekatan perataan berbasis koridor keselamatan baru menggunakan. *Bezier curve*.

Tahapan pertama pada penelitian ini yaitu membuat sebuah lingkungan 3D yang akan dilalui oleh *Fixed wing*, kemudian algoritma *Visibility Roadmap* dipakai sebagai pembuatan *node* dan *edge* yang akan digunakan oleh algoritma Theta\* untuk mencari jalur terpendek setelah diberikan koordinat awal dan tujuan, kemudian sebuah proses optimisasi memastikan kelancaran jalur dengan mengaplikasikan satu set *Bezier curve* ke jalur awal, untuk pembuatan *Bezier curve* pada penelitian ini akan mempertimbangkan manuver dari *Fixed wing* sebagai syarat dalam pembuatan jalur baru.

Kata kunci: Theta\*, *Fixed wing*, *Bezier curve*, Manuver, *Visibility roadmap*

*Halaman ini sengaja dikosongkan*

## DAFTAR ISI

LEMBAR PENGESAHAN .....	pii
ABSTRAK .....	v
DAFTAR ISI.....	vii
DAFTAR GAMBAR .....	ix
DAFTAR TABEL.....	xi
BAB 1 PENDAHULUAN .....	1
1.1 Latar Belakang .....	1
1.2 Rumusan Masalah.....	2
1.3 Tujuan .....	p2
1.4 Batasan Masalah .....	2
1.5 Kontribusi .....	2
BAB 2 KAJIAN PUSTAKA.....	3
2.1 Kajian Penelitian Terkait .....	3
2.1.1. Path planning Strategies for UAVs in 3D Environment [1].....	3
2.1.2. Efficient Energy Flight Path planning Algoritma Using 3-D Visibility roadmap for Small Unmanned Aerial Vehicle [2] .....	12
2.1.3. Filtered Medial Surface Based Approach for 3D Collision-Free Path Planning Problem [3] .....	15
2.2 Teori Dasar.....	18
2.2.1 Fixed Wing UAV [4].....	18
2.2.2 Algoritma Theta* [1].....	26
2.2.3 Metode State Dependent-Linier Quadratic [6].....	26
BAB 3 METODOLOGI PENELITIAN.....	33
3.1 Memodelkan Lingkungan 3D .....	33
3.1.1 Memodelkan Digital Map dan Convex Obstacle .....	33
3.1.2 Algoritma pembuatan Roadmap .....	34
3.2 Theta* dan Pencarian Path Terpendek.....	36
3.3 Fixed Wing Ultrastick-25 (Manuever).....	37
3.4 Strategi Smoothing menggunakan Bezier curve.....	39
3.5 Perancangan Kontrol Fixed Wing.....	40

3.6	Diagram Blok Keseluruhan.....	41
3.7	Hipotesa Penelitian .....	42
3.8	Rencana Pengujian.....	42
3.9	Kriteria Pengujian .....	42
BAB 4 RENCANA DAN JADWAL KEGIATAN .....		43
4.1	Rencana Kegiatan .....	43
4.2	Jadwal Kegiatan .....	44
DAFTAR PUSTAKA .....		47



## DAFTAR GAMBAR

Gambar 2.1 Path 1 3D representation .....	4
Gambar 2.2 Path 1 3D representation orographic obstacle (Longitude-latitude plane).....	5
Gambar 2.3 Path 1 3D representation orographic obstacle (Flight altitude).....	5
Gambar 2.4 Path 2 3D representation orographic obstacle.....	6
Gambar 2.5 Path 2 3D representation orographic obstacle (Longitude-latitude plane).....	7
Gambar 2.6 Path 2 3D representation orographic obstacle (Flight Altitude) .....	7
Gambar 2.7 Path 1 3D Urban environment.....	8
Gambar 2.8 Path 1 3D Urban environment (X-Y plane) .....	9
Gambar 2.9 Path 1 3D Urban environment (Flight Altitude) .....	9
Gambar 2.10 Path 2 Urban environment.....	10
Gambar 2.11 Path 2 3D Urban environment (X-Y plane) .....	11
Gambar 2.12 Path 2 3D Urban environment (Flight Altitude) .....	11
Gambar 2.13 Roadmap pada Single Obstacle.....	13
Gambar 2.14 Visualisasi pada 10 obstacle.....	14
Gambar 2.15 Simulasi dari 2 metode untuk semua variasi obstacle.....	14
Gambar 2.16 FMS berdasarkan ukuran UAV (a) $d = 20cm$ (b) $d = 40 cm$ .....	16
Gambar 2.17 Path terpendek pada FMS dengan dimensi UAV yang berbeda a) $20 cm$ b) $40 cm$ dengan altitude plot untuk masing-masing kasus. ....	16
Gambar 2.18 Perbandingan path sebelum dan setelah proses smoothing dengan .....	17
ukuran UAV yang berbeda (ungu dan biru, Bezier curve) (oren dan .....	17
Gambar 2.19 Vehicle Frame .....	19
Gambar 2.20 Vehicle-1 frame.....	19
Gambar 2.21 Vehicle-2 frame.....	20
Gambar 2.22 Body frame.....	20
Gambar 2.23 Stability frame .....	21
Gambar 2.24 Wind frame.....	22
Gambar 2.25 Wind triangle secara horizontal.....	23
Gambar 2.26 Wind triangle secara vertikal.....	23
Gambar 3.1 a) Contoh Digital Map, b) Convex Obstacle 3D.....	34
Gambar 3.2 Flowchart perancangan algoritma roadmap .....	35
Gambar 3.3 Flowchart algoritma Theta* .....	37
Gambar 3.4 Skema kontrol Fixed Wing .....	40
Gambar 3.5 Blok Diagram Sistem Keseluruhan .....	41

*Halaman ini sengaja dikosongkan*

## DAFTAR TABEL

Tabel 2.1 Path 1 <i>Orographic Obstacle</i> .....	6
Tabel 2.2 Path 2 <i>Orographic Obstacle</i> .....	8
Tabel 2.3 Path 1 <i>Urban environment</i> .....	10
Tabel 2.4 Path 2 3D <i>Urban environment</i> .....	11
Tabel 2.5 Jumlah <i>obstacle</i> dan waktu <i>processing</i> .....	13
Tabel 2.6 Perbandingan Metode Roadmap dan Grid .....	15
Tabel 2.7 Perbandingan kuantitatif antara path A* dan <i>Bezier cruve</i> smoothing....	17
Tabel 2.8 Parameter <i>Fixed Wing</i> .....	24
Tabel 2.9 Parameter pada dinamika <i>Fixed Wing</i> .....	24
Tabel 4.1 Jadwal Kegiatan .....	44

*Halaman ini sengaja dikosongkan*

# BAB 1

## PENDAHULUAN

### 1.1 Latar Belakang

Perencanaan jalur adalah salah satu teknologi yang memungkinkan robot untuk berhasil melintasi lingkungan yang berantakan. Sehingga telah banyak penelitian yang membahas tentang permasalahan tersebut, seperti yang telah dibahas pada [1] yang membandingkan 2 buah algoritma heuristik yaitu A\* dan Theta\* untuk menemukan jalur terpendek untuk *Unmanned Aerial Vehicle* (UAV) di sebuah lingkungan yang telah dibuat oleh peneliti dan membuktikan bahwa algoritma Theta\* memiliki hasil yang bagus untuk mencari jalur terpendek dibandingkan dengan algoritma A\*, namun penulis mengujikan kelayakan algoritma ini dalam lingkungan *obstacle* yang sederhana, oleh karena itu untuk membuat lingkungan *obstacle* dengan skala yang besar dan memiliki resolusi tinggi maka digunakan metode *visibility graph* yang disebut *visibility roadmap* seperti yang telah dilakukan oleh [2] yang memodelkan data ketinggian peta dengan satu set *convex obstacle* dengan menghitung *convex hull* dan ketika UAV melakukan pencarian jalur di dalam lingkungan yang penuh hambatan ini metode *visibility graph* ini juga efektif dalam penghindaran *obstacle*. Metode *visibility graph* ini juga memiliki kelebihan yaitu dapat membawa keuntungan untuk kontrol konsumsi energi seperti yang telah dilakukan oleh [2] dengan menambahkan algoritma *bounded space* pada *visibility roadmap*.

Pada penelitian [2] juga tidak mempertimbangkan kinematika dari robot sehingga membutuhkan proses *smoothing* untuk merealokasi urutan *node* untuk mendapatkan jalur yang dapat diterbangkan. Sebuah solusi, yang diadopsi untuk memperlancar jalur sesuai dengan radius belok dan tingkat batasan pendakian adalah penggunaan *Bezier curve*. Ini adalah solusi saat ini yang diadopsi sebagai post-smoother dalam perencanaan jalur yang dikembangkan oleh [3] .

## 1.2 Rumusan Masalah

Rumusan masalah dari penelitian ini adalah bagaimana merancang perencanaan jalur pada *Fixed Wing* untuk menemukan jalur terpendek pada *visibility roadmap* dengan menggunakan algoritma Theta\* dan jalur yang telah dihaluskan menggunakan algoritma *Bezier curve*.

## 1.3 Tujuan

Tujuan dari penelitian ini adalah *Map* yang disajikan menghasilkan semua *node* dan *edge* yang dibutuhkan oleh Theta\* untuk mendapatkan jalur terpendek dan jalur yang dihaluskan dengan *Bezier Curve* (berdasarkan kemampuan manuver) sehingga jalur tersebut dapat di-tracking oleh *Fixed Wing*.

## 1.4 Batasan Masalah

Dalam penelitian ini terdapat batasan masalah untuk membatasi permasalahan yang muncul diantaranya,

1. *Fixed Wing* yang digunakan adalah tipe *Fixed Wing* ultrastick-5.
2. *Urban environment* direpresentasikan sebagai blok-blok persegi.
3. Pemilihan jalur tidak memperhitungkan konsumsi energi dari *Fixed Wing*.
4. Bangunan/*obstacle* yang dibuat untuk merepresentasikan *urban environment* adalah 20, 30 dan 40 bangunan.

## 1.5 Kontribusi

Kontribusi dari penelitian ini adalah menghasilkan jalur yang halus sesuai dengan ketentuan manuver dari *Fixed Wing* Ultrastick-25.

## **BAB 2**

### **KAJIAN PUSTAKA**

Pada sub bab ini akan membahas kajian penelitian terkait yang telah dilakukan dan menjadi referensi penulis untuk membuat tesis yang akan dilakukan. Selain itu, juga terdapat penjelasan tentang metode atau semua teori yang akan digunakan untuk keperluan penulis dalam pembuatan tesisnya.

#### **2.1 Kajian Penelitian Terkait**

Pada penyusunan tesis ini, dilakukan kajian terhadap penelitian-penelitian terkait yang telah ada. Berbagai metode dan struktur kontrol yang digunakan pada setiap pustaka akan dipaparkan dalam sub-bab ini, sehingga diperoleh ide dasar untuk penyusunan tesis.

##### **2.1.1. Jalur *planning* Strategies for UAVs in 3D Environment [1]**

*Paper* ini membahas tentang strategi jalur *planning* dan membandingkan 2 buah algoritma heuristik yaitu Theta\* (*basic version*) dan A\* untuk lingkungan 3D, lalu kedua algoritma tersebut diujikan pada dua macam lingkungan statis yaitu *orographic obstacle* dan lingkungan perkotaan (*urban environment*). Pemodelan lingkungan 3D menggunakan alat perencanaan berbasis MATLAB yaitu, *Digital Elevation Maps* (DEMs) untuk meminimalkan benturan pada lingkungan *orographic*.

Algoritma A \* secara iteratif mengevaluasi biaya pemindahan dari sel saat ini ke salah satu tetangganya melalui fungsi biaya yang ditentukan. Fungsi ini ( $F$ ) diperoleh dengan menambahkan dua buah istilah:

- $H$  proporsional dengan jarak estimasi heuristik dari sel yang dievaluasi ke tujuan.
- $G$  proporsional dengan jarak dari sel saat ini ke yang dievaluasi.

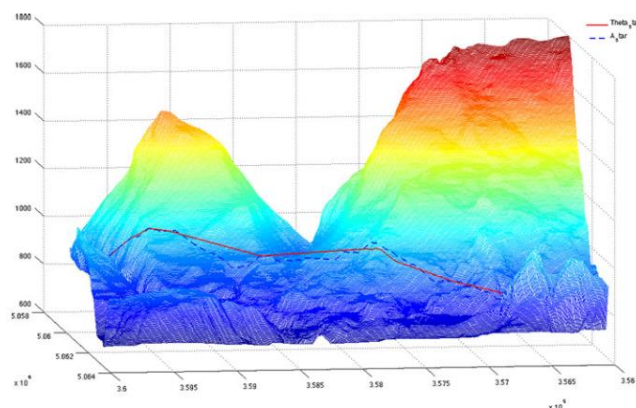
Nilai  $G$  adalah 0 untuk sel awal dan akan meningkat ketika algoritma memperluas sel-sel (misal, pada setiap langkah algoritma menjumlahkan biaya pemindahan dari sel awal ke sel saat ini, jarak dari sel saat ini ke sel tetangganya.

Salah satu kelemahan terpenting dari algoritma A\* berada pada *heading constraints* yang terhubung dengan karakteristik *grid*. Beberapa algoritma telah dikembangkan dari algoritma A\* salah satunya adalah Theta\*.

Theta\* merupakan salah satu algoritma yang menyempurnakan pencarian grafik yang mendapatkan jalur umum dengan *heading* apa saja. Diketahui bahwa algoritma pencarian grafik ini memilih langkah per langkah dari satu *node* ke *node* berikutnya. Menentukan *parent* dari *node* sebelumnya, hanya sampai posisi saat ini, dan *neighbour node* dievaluasi untuk langkah selanjutnya.

Pada saat algoritma Theta\* memperluas pencarian, ini menghasilkan 2 tipe jalur yaitu dari *node* saat ini ke *neighbour* (seperti pada A\*) dan dari *parent node* saat ini ke *neighbour*. Dibandingkan dengan A\*, *parent* dari sebuah *node* tidak harus menjadi *neighbour* dari *node* tersebut selama ada garis pandang (LoS) yang menghubungkan antara kedua *node* tersebut, sehingga jalur yang ditemukan oleh Theta\* merupakan solusi untuk menemukan jalur terpendek dan lebih mulus dari algoritma A\*.

Kedua algoritma ini akan diujikan pada 2 lingkungan 3D yaitu *orographic obstacle* yang dibuat menggunakan peta DEMs dan *urban environment* (lingkungan perkotaan) menggunakan GUI di MATLAB . Dua jalur di dataran tinggi dibuat untuk *orographic obstacle* yang pertama jalur dengan jalur jarak menengah yang ditunjukkan pada gambar 2.1 dan yang kedua sebuah pemisahan rintangan orografis (*orographic obstacle separation*) yang ditunjukkan pada gambar 2.4.



Gambar 2.1 Jalur 1 3D *representation*



Dengan karakteristik map

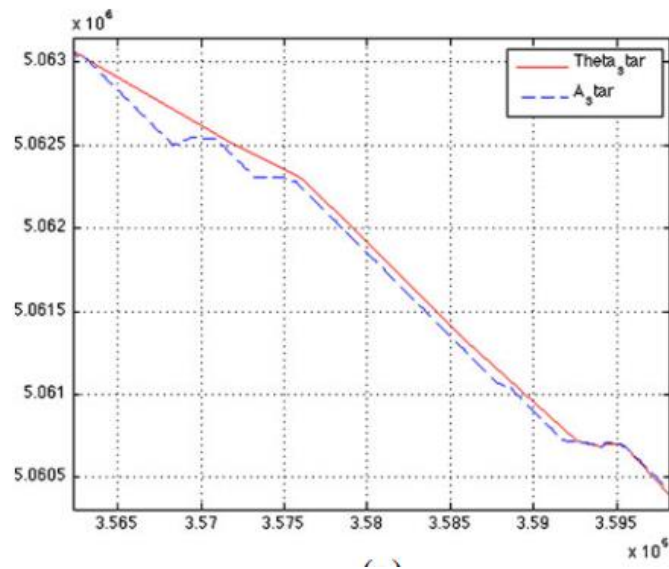
Jumlah node : 7.634,088

$\Delta\text{lat}$  : 10 m

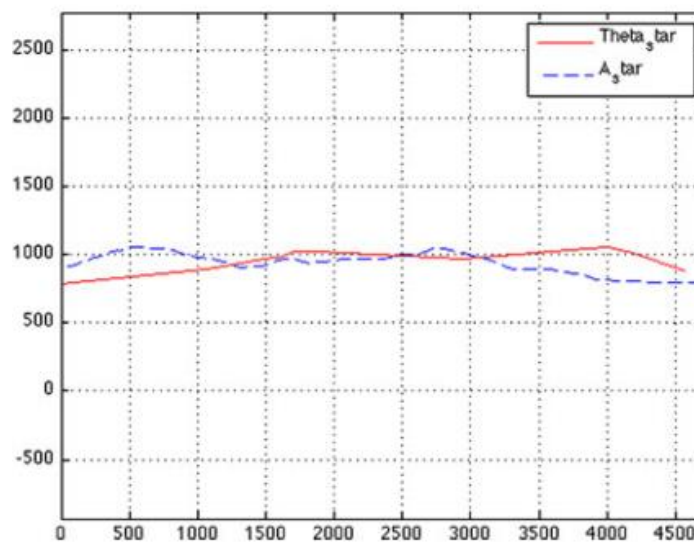
$\Delta\text{long}$  : 10 m

$\Delta Z$  : 5 m

Dimensi matriks lingkungan :  $357 \times 396 \times 54$  ( $\text{lat} \times \text{long} \times Z$ )



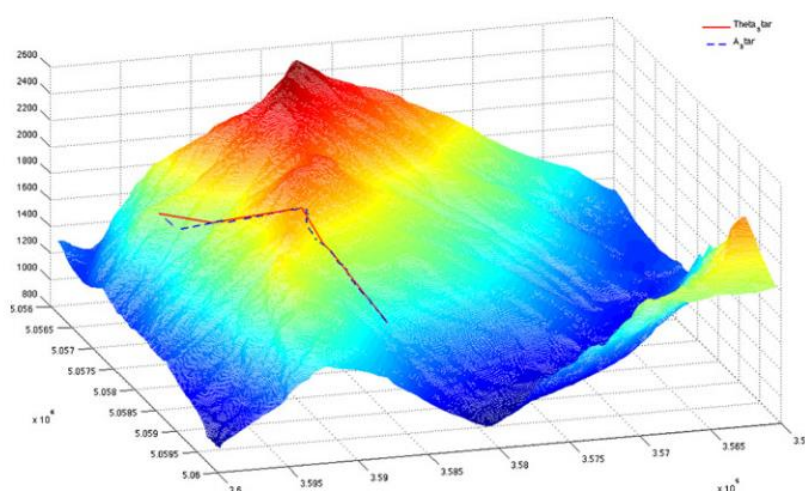
Gambar 2.2 Jalur 1 3D representation orographic obstacle (Longitude-latitude plane)



Gambar 2.3 Jalur 1 3D representation orographic obstacle (Flight altitude)

Tabel 2.1 Jalur 1 Orographic Obstacle

<b>Jalur 1</b>	<b>A*</b>	<b>Theta*</b>
Panjang jalur (m)	4850	4618
Waktu komputasi (s)	1.203	1.393
Jumlah perubahan <i>heading</i>	42	13
Jumlah perubahan ketinggian	159	15
Jumlah poin jalur	258	17



Gambar 2.4 Jalur 2 3D *representation orographic obstacle*

Dengan karakteristik map

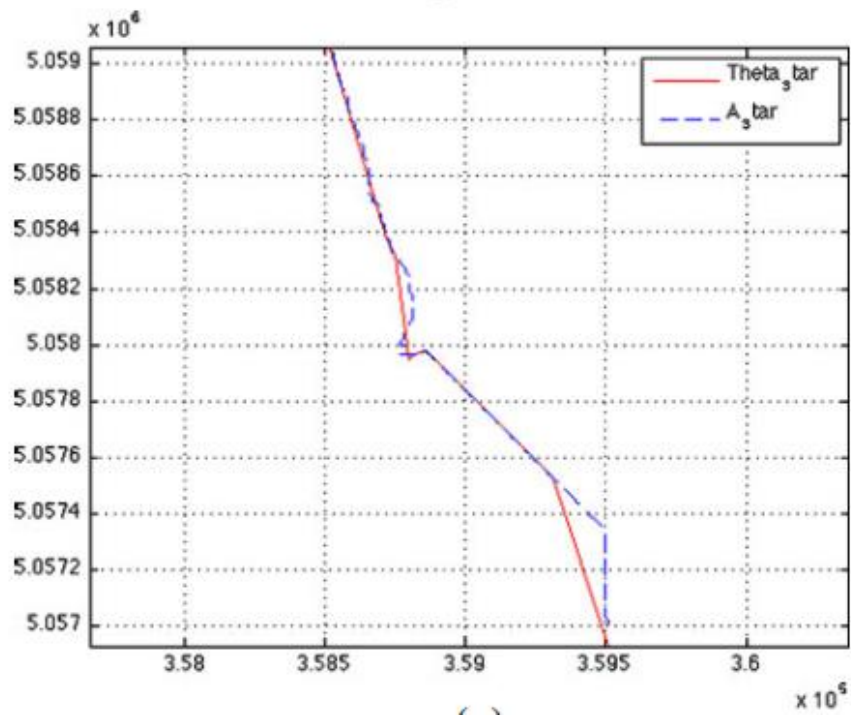
Jumlah node : 16.727,040

$\Delta lat$  : 10 m

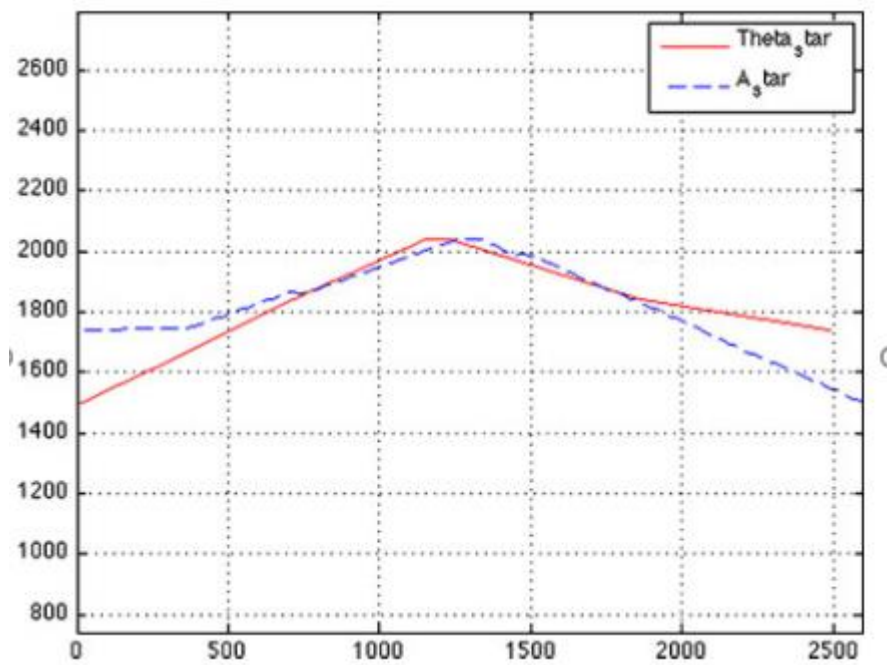
$\Delta long$  : 10 m

$\Delta Z$  : 5 m

Dimensi matriks lingkungan :  $384 \times 396 \times 110$  (*lat*  $\times$  *long*  $\times$  *Z*)



Gambar 2.5 Jalur 2 3D representation orographic obstacle obstacle (Longitude-latitude plane)

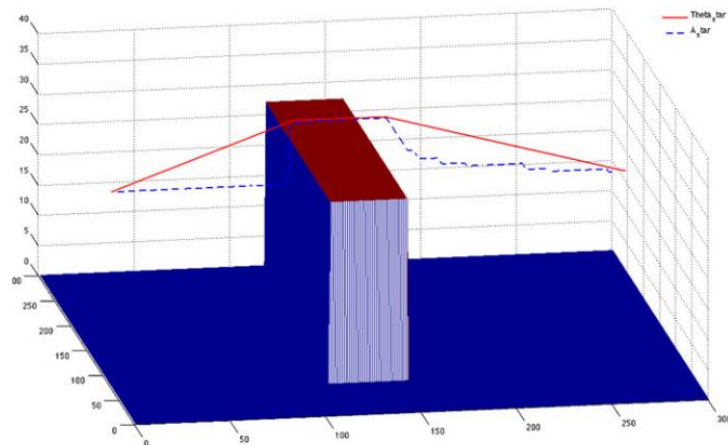


Gambar 2.6 Jalur 2 3D representation orographic obstacle (Flight Altitude)

Tabel 2.2 Jalur 2 *Orographic Obstacle*

<b>Jalur 2</b>	<b>A*</b>	<b>Theta*</b>
Panjang jalur (m)	2776	2653
Waktu komputasi (s)	1.622	1.638
Jumlah perubahan <i>heading</i>	66	9
Jumlah perubahan ketinggian	174	8
Jumlah poin jalur	220	11

Pengujian algoritma pada *urban environment* juga dibuat dalam 2 bentuk jalur. Jalur pertama menempatkan satu bangunan ditengah jalur pencarian dapat dilihat pada gambar 2.7 sedangkan jalur kedua dibuat sebuah lingkungan dengan beberapa bangunan dapat dilihat pada gambar 2.10. Pemodelan lingkungan untuk *urban environment* ini dibuat sangat sederhana oleh peneliti tetapi berguna untuk menguji algoritma dengan peta yang meniru karakteristik dari lingkungan berantakan.



Gambar 2.7 Jalur 1 3D *Urban environment*

Dengan karakteristik map

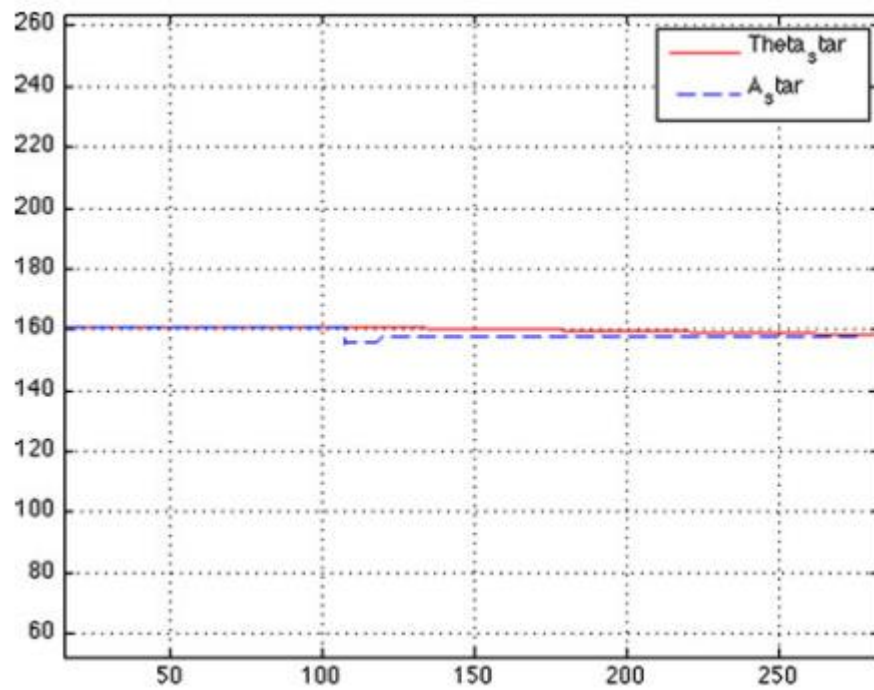
Jumlah node : 9,990,000

$\Delta X$  : 1 m

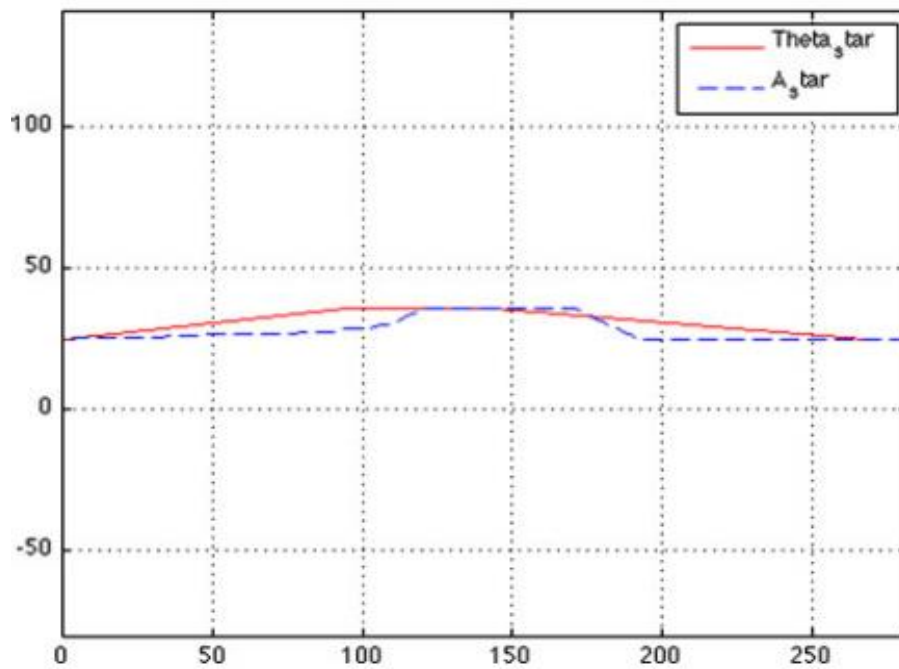
$\Delta Y$  : 1 m

$\Delta Z$  : 0.5 m

Dimensi matriks lingkungan :  $300 \times 300 \times 111$  (*lat*  $\times$  *long*  $\times$  *Z*)



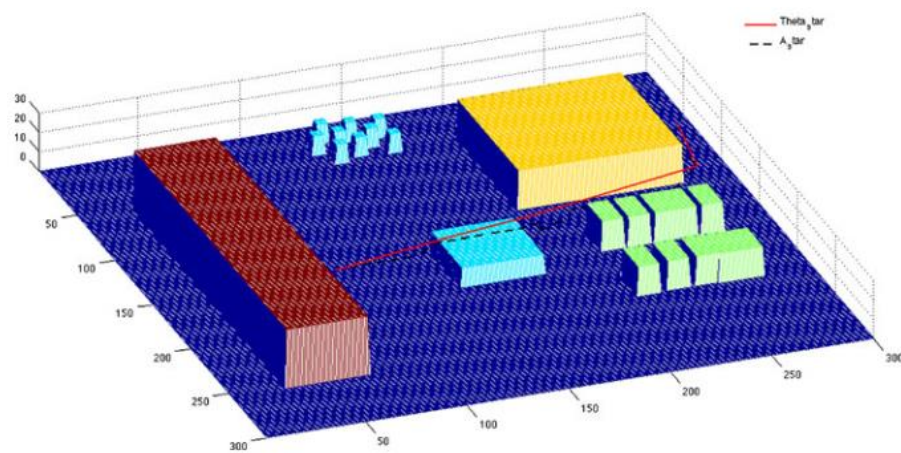
Gambar 2.8 Jalur 1 3D *Urban environment* (X-Y plane)



Gambar 2.9 Jalur 1 3D *Urban environment* (Flight Altitude)

Tabel 2.3 Jalur 1 *Urban environment*

<b>Urban Jalur 1</b>	<b>A*</b>	<b>Theta*</b>
Panjang jalur (m)	287	269
Waktu komputasi (s)	5.718	3.081
Jumlah perubahan <i>heading</i>	15	2
Jumlah perubahan ketinggian	42	2
Jumlah poin jalur	282	4



Gambar 2.10 Jalur 2 *Urban environment*

Dengan karakteristik map

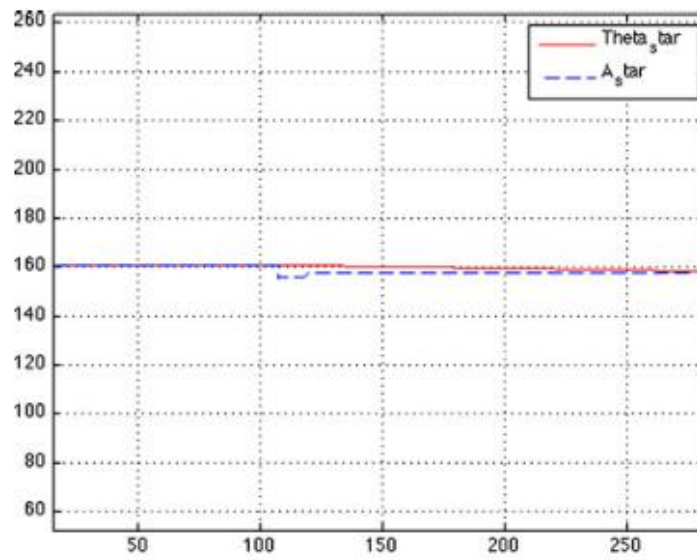
Jumlah titik : 152,064

$\Delta X$  : 1 m

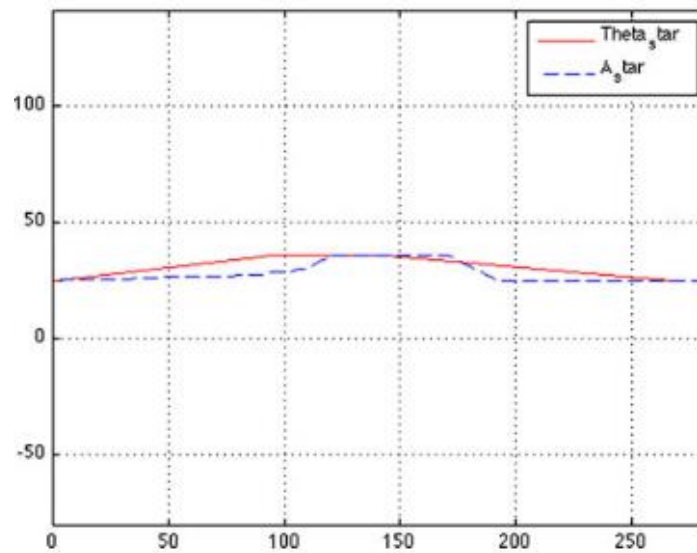
$\Delta Y$  : 1 m

$\Delta Z$  : 0.5 m

Dimensi matriks lingkungan :  $300 \times 300 \times 111$  (*lat*  $\times$  *long*  $\times$  *Z*)



Gambar 2.11 Jalur 2 3D *Urban environment* (X-Y plane)



Gambar 2.12 Jalur 2 3D *Urban environment* (Flight Altitude)

Tabel 2.4 Jalur 2 3D *Urban environment*

<b>Urban Jalur 1</b>	<b>A*</b>	<b>Theta*</b>
Panjang jalur (m)	287	269
Waktu komputasi (s)	5.718	3.081
Jumlah perubahan <i>heading</i>	15	2
Jumlah perubahan ketinggian	42	2
Jumlah poin jalur	282	4



Dari hasil pengujian yang telah dilakukan peneliti menyimpulkan bahwa algoritma Theta\* memiliki hasil yang lebih baik dan dapat menemukan jalur terpendek dan perubahan *heading* yang kecil dibandingkan algoritma A\*. Peneliti mengatakan bahwa *paper* ini memiliki beberapa kekurangan yaitu pertama, peneliti tidak mempertimbangkan kinematika UAV sebagai bagian dalam jalur *generation* dan ini menjadi masalah utama untuk kendaraan *nonholonomic* yang membutuhkan proses perataan untuk merealokasi urutan *node* untuk mendapatkan jalur yang dapat diterbangkan. Oleh karena itu peneliti menyarankan menambahkan sebuah algoritma untuk *safety* dan *smooth* untuk penerbangan. Kedua, karena pengecekan algoritma pada metode *mesh grid* untuk bangunan yang terpisah lebih sulit oleh karena itu pengujian pada *urban environment* dibuat lebih sederhana karena hanya untuk menguji kemampuan algoritma. Sehingga peneliti menyarankan untuk mengaplikasikan algoritma ini pada *urban environment*.

Ide yang didapat dari *paper* ini yaitu menggunakan algoritma Theta\* untuk menemukan jalur terpendek pada *urban environment*.

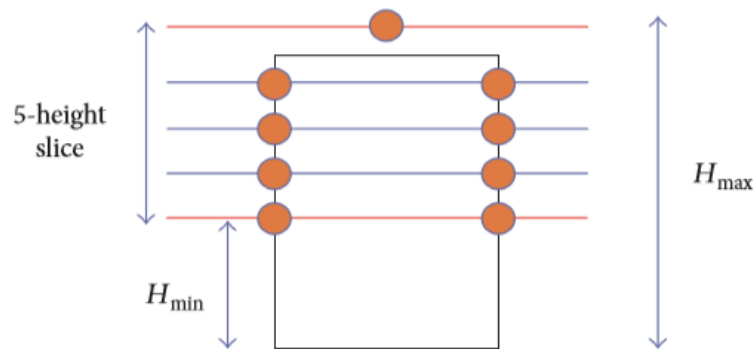
### **2.1.2. Efficient Energy Flight Jalur *planning* Algoritm Using 3-D Visibility roadmap for Small Unmanned Aerial Vehicle [2]**

*Paper* ini menyajikan algoritma jalur *planning* dalam lingkungan 3D dengan *obstacle convex* statis untuk *Small Unmanned Aerial Vehicle* (SUAV) *Fixed Wing*. Jalur *planning* yang digunakan pada *paper* ini menggunakan metode *visibility roadmap* berdasarkan pada *visibility graph* untuk menentukan semua jalur yang dapat dilalui oleh SUAV dari titik awal ke titik tujuan yang telah ditetapkan dan memperhitungkan konsumsi energi yang dibutuhkan dalam pemilihan jalur .

*Paper* ini membuat 2 tahap prosedur yaitu pertama fase *preprocessing*, pada fase ini membuat *visibility graph* penuh yang disebut *visibility roadmap* yang menyediakan setiap koneksi *node* yang mungkin dilalui. Kedua, fase, pada pencarian dimulai ketika modul pencarian jalur menerima koordinat awal dan koordinat tujuan. Dengan menghubungkan koordinat ini ke *roadmap*, dan memperoleh jalur penerbangan yang dioptimalkan menggunakan algoritma A\* tepat setelah misi dimulai.



Untuk membangun *roadmap*, langkah pertama melibatkan pengambilan sampel 3D untuk *node* yang memungkinkan. Peta ini dibagi menjadi k-layer oleh bidang horizontal k dengan jarak  $d_{cut}$  yang sama dari  $H_{min}$  ke  $H_{max}$ . Untuk menemukan *node* yang terletak di dalam batas *obstacle* dengan menghitung titik persimpangan bidang potong dan tepi *obstacle*.

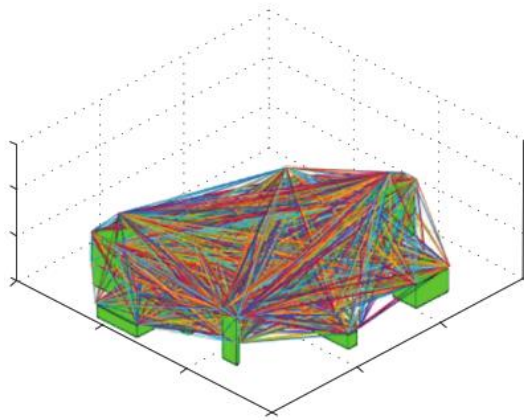


Gambar 2.13 *Roadmap* pada *Single Obstacle*

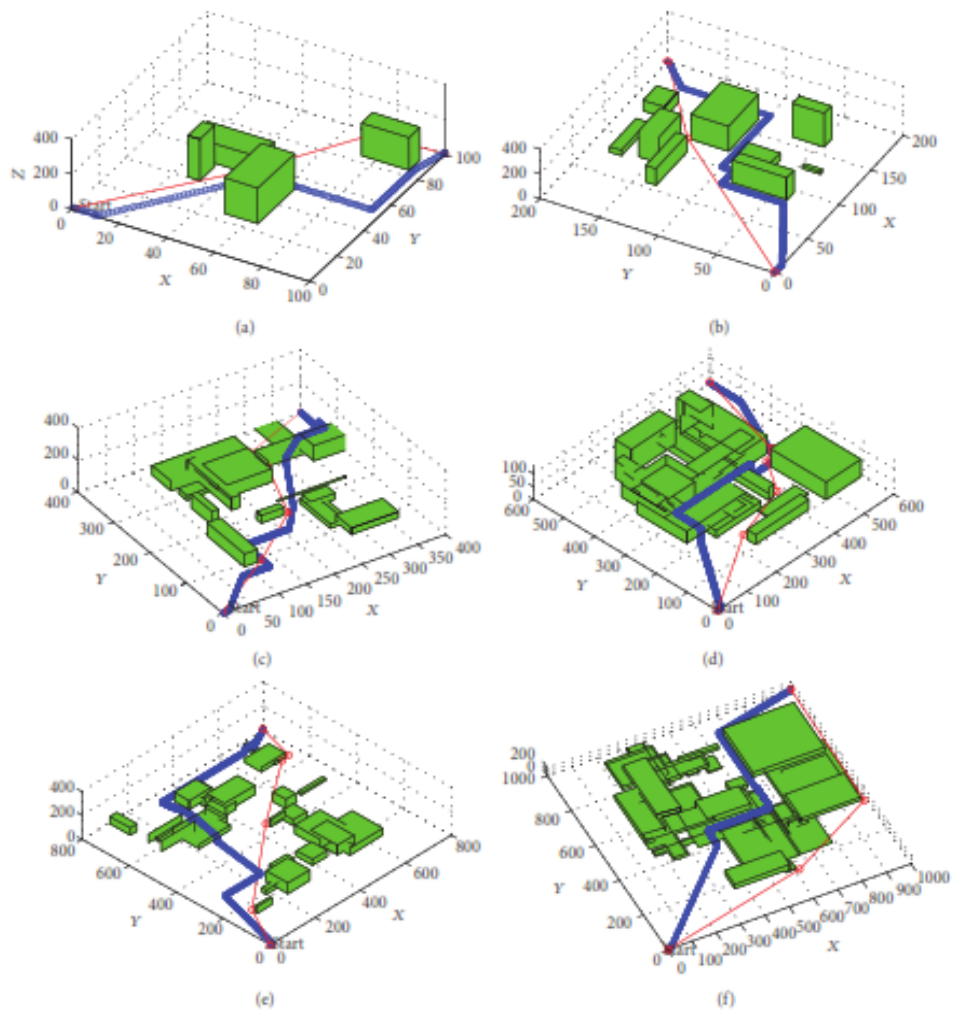
Gambar diatas menjelaskan *visibility roadmap* membuat *single obstacle*, *Paper* ini membuat lingkungan 3D dengan beberapa jumlah *obstacle* dapat dilihat pada tabel 2.5.

Tabel 2.5 Jumlah *obstacle* dan waktu *processing*

Jumlah <i>Obstacle</i>	Waktu Prosesing (s)
5	1.4
10	11.4
15	26.7
20	53.8
25	101
30	133



Gambar 2.14 Visualisasi pada 10 *obstacle*



Gambar 2.15 Simulasi dari 2 metode untuk semua variasi *obstacle*

Tabel 2.6 Perbandingan Metode *Roadmap* dan *Grid*

Number of tests	Proposed visibility roadmap			Grid method		
	Energy	Distance	Heading changes	Energy	Distance	Heading changes
5	38,246	879.8	2.4	42,050	905.2	6.8
10	37,945	872.6	2.8	41,895	902.9	7.8
20	37,905	866.2	3.1	47,678	938.3	8
30	37,746	868	3.7	41,690	898.3	7.5
40	38,724	879.2	2.9	42,530	911.9	8.5
50	38,507	877.8	3.1	43,684	928.1	9.1

Pada paper ini peneliti membandingkan hasil dari penelitiannya dengan tujuan penelitian yang sama tetapi dengan menggunakan metode *grid*, dapat dilihat pada tabel 2.6 bahwa *visibility roadmap* memberikan hasil terbaik dari metode *grid*.

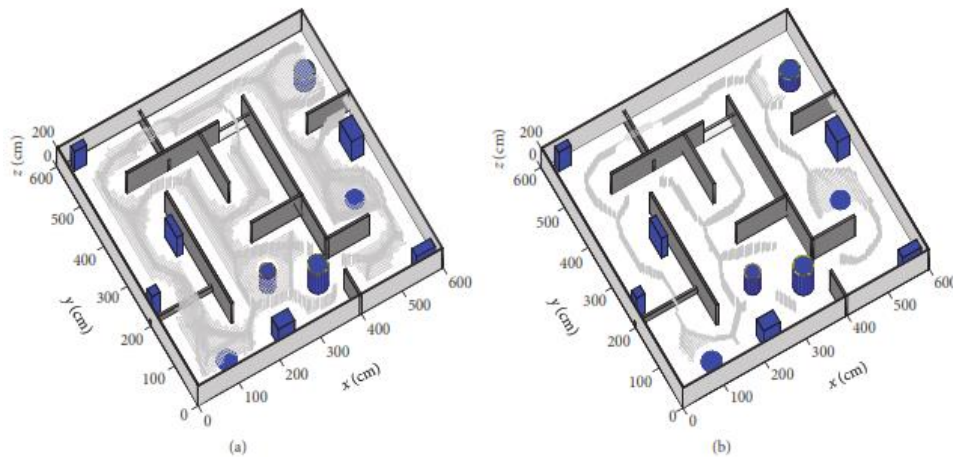
Ide yang dapat diambil dari penelitian ini adalah menggunakan metode *visibility graph* untuk pembuatan peta lingkungan 3D untuk menyediakan *node* dan *edge* untuk menentukan jalur terpendek yang akan di lalui oleh *Fixed Wing* UAV.

### 2.1.3. Filtered Medial Surface Based Approach for 3D Collision-Free Path Planning Problem [3]

Penelitian yang dilakukan oleh Karima Benzaid, Romain Marie, Noura Mansouri, dan Ouiddad Labbani-Igbida ini memperhatikan kemulusan jalur dan jalur bebas tabrakan untuk UAV saat melintasi jalur yang telah dipilih untuk mencapai titik tujuan, Peneliti memperkenalkan algoritma skeletonisasi 3D untuk membuat lingkungan yang akan dilalui oleh UAV berdasarkan algoritma representasi bentuk 2D atau disebut sebagai FMS (*Filtered Medial Surface*) yang memperhitungkan dimensi dari UAV.

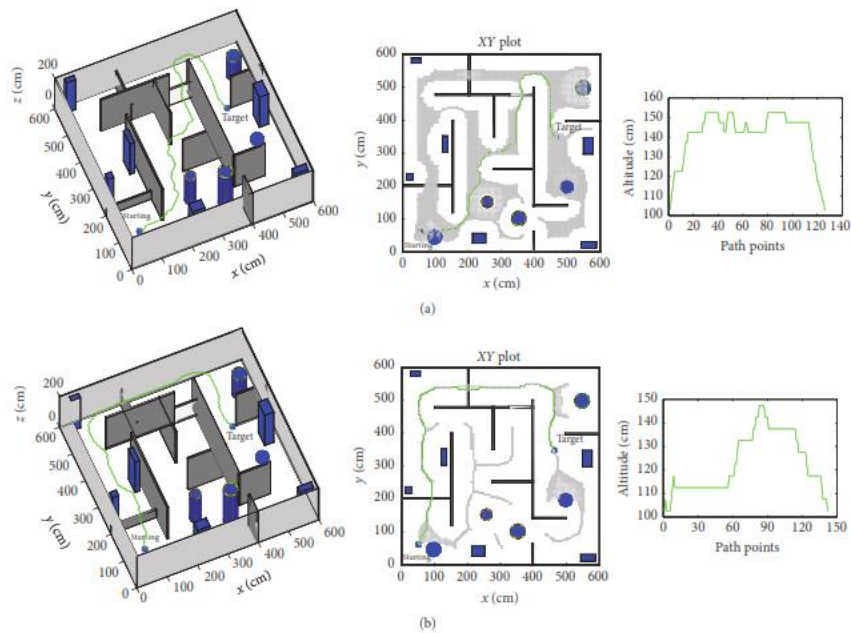
Algorithma A\* diterapkan sebagai algorithma penemuan jalur terpendek pada FMS. Untuk menghilangkan osilasi dan belokan yang kencang di jalur yang dihasilkan oleh A\*, peneliti mengusulkan pendekatan perataan berbasis koridor keselamatan baru menggunakan *Bezier curve*. Kemudian Peneliti melakukan studi komparatif dari pendekatan yang diusulkan (*Bezier curve*) dengan algorithma A\*

secara langsung pada FMS dengan serangkaian kriteria kuantitatif. Pada gambar 2.16 dibuat lingkungan 3D untuk 2 macam dimensi UAV.



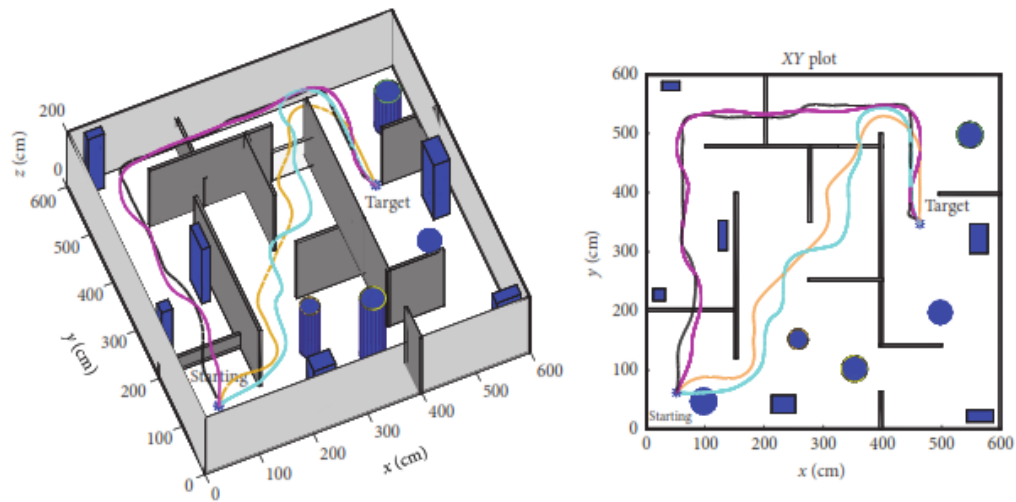
Gambar 2.16 FMS berdasarkan ukuran UAV (a)  $d = 20\text{ cm}$  (b)  $d = 40\text{ cm}$

Setelah lingkungan 3D dibuat di FMS kemudian algoritma A\* mencari jalur terpendek yang akan dilalui UAV pada FMS. Penemuan jalur oleh A\* dapat dilihat pada gambar 2.17.



Gambar 2.17 Jalur terpendek pada FMS dengan dimensi UAV yang berbeda a)  $20\text{ cm}$  b)  $40\text{ cm}$  dengan altitude plot untuk masing-masing kasus.

Jalur yang dihasilkan harus dilalui oleh UAV secara elegan dan tenang tanpa adanya osilasi, namun jalur yang dihitung pada FMS belum tentu mulus. Oleh karena itu untuk mengatasi masalah ini, peneliti mengusulkan langkah pemrosesan tambahan berdasarkan penggunaan 3D *Bezier curve*, sehingga dapat dilihat pada gambar 2.18 membandingkan jalur sebelum dan setelah menambahkan proses smoothing. Hasil dai kriteria kuantitatif yang diberikan pada tabel 2.7.



Gambar 2.18 Perbandingan jalur sebelum dan setelah proses smoothing dengan ukuran UAV yang berbeda (ungu dan biru, *Bezier curve*) (oren dan hitam,A \*)

Tabel 2.7 Perbandingan kuantitatif antara jalur A\* dan *Bezier cruve smoothing*

	A-star on the OG		Proposed approach	
	case 1	case 2	case 1	case 2
Input data number	236644	61841	23324	7913
Path length (m)	9.03	11.08	9.95	11.41
Number of path points	155	209	126	142
Smallest dist. to obstacles (cm)	25.63	42.86	31.13	46.28
Mean of dist. to obstacles (cm)	40.18	50.51	48.42	56.88

Dari hasil pengujian peneliti, dapat disimpulkan bahwa metode *Bezier curve* mampu membuat jalur lebih halus dan bebas tabrakan,oleh karena itu penulis

mengajukan metode ini sebagai pra *smoothing* jalur setelah mendapatkan jalur terpendek menggunakan algoritma Theta\*.

## 2.2 Teori Dasar

Pada bab ini terdapat teori dasar yang menunjang dalam merumuskan dan menyelesaikan masalah yang dihadapi dalam mengerjakan tesis. Bagian awal terdapat teori tentang quadcopter secara umum dan konsep gerak dari quadcopter. Bagian selanjutnya membahas tentang metode kontrol gerak quadcopter.

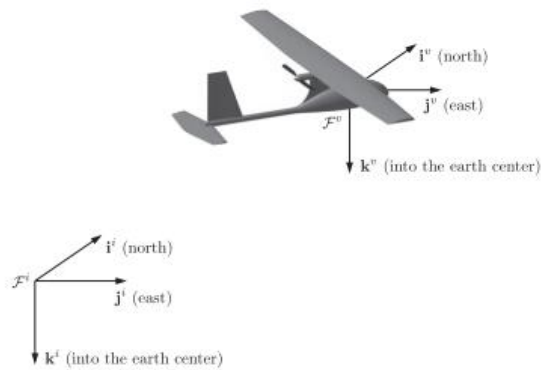
### 2.2.1 *Fixed Wing* UAV [4]

Pada pemodelan *Fixed Wing*, akan dilakukan analisa dari *frame* koordinat UAV, *wind triangle*, parameter yang diperlukan pada pemodelan *Fixed Wing*, kinematika dan dinamika *Fixed Wing*.

#### Frame Koordinat UAV

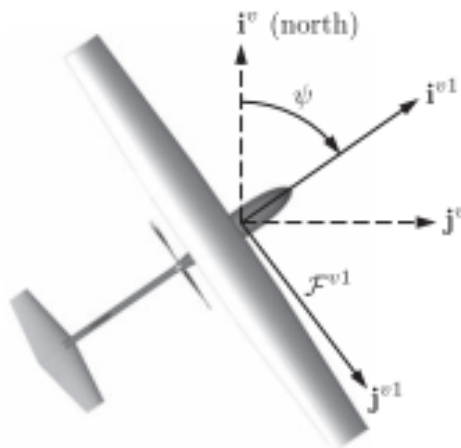
Untuk menentukan arah vektor dari UAV maka digunakan koordinat Cartesian. Pada bagian ini membahas berbagai koordinat pada sistem yang digunakan pada UAV dimulai dari posisi dan orientasi UAV dan sensor, dan transformasi antara koordinat sistem. Pada setiap koordinat memiliki inisialisasi yang berbeda. Koordinat frame yang digunakan:

- *Inertial frame* ( $f^i$ ) : *Frame* ini adalah *frame* dari bumi dan biasa disebut *North-East-Down (NED) frame*.
- *Vehicle frame* ( $f^v$ ) : Pusat dari *vehicle frame* adalah pusat massa UAV. Sumbu pada *vehicle frame* sesuai dengan sumbu *inertial frame*.



Gambar 2.19 *Vehicle Frame*

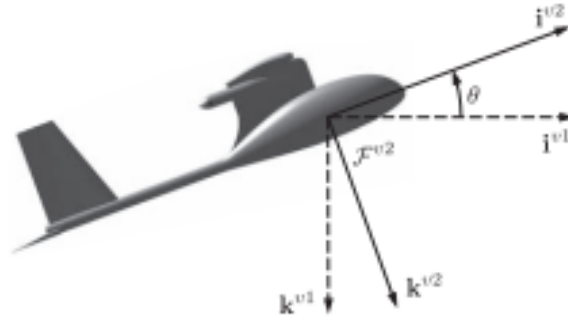
- *Vehicle-1 frame* ( $F^{v1}$ ) : Pada *frame vehicle-1* adalah rotasi UAV pada sudut ( $\Psi$ ) dengan menggunakan kaidah tangan kanan.



Gambar 2.20 *Vehicle-1 frame*

dimana,  $i^{v1}$  adalah arah hidung UAV,  $j^{v1}$  adalah arah sumbu sayap kanan, sedangkan  $k^{v1}$  bersesuaian dengan  $k^v$  dan arah menuju bumi.

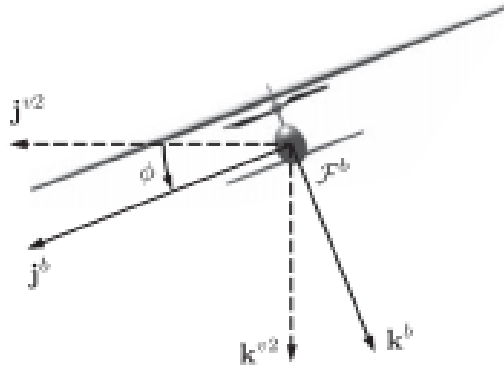
- *Vehicle-2 frame* ( $F^{v2}$ ) : Pada *frame vehicle-2* adalah rotasi UAV pada sudut ( $\theta$ ) dengan menggunakan kaidah tangan kanan



Gambar 2.21 *Vehicle-2 frame*

dimana,  $i^{v2}$  adalah arah hidung UAV,  $j^{v2}$  adalah arah sumbu sayap kanan, sedangkan  $k^{v2}$  merupakan arah tegak lurus UAV menuju bumi.

- *Body frame* ( $\mathcal{F}^b$ ) : Rotasi *body frame* terhadap sudut *roll* ( $\phi$ ) dengan menggunakan kaidah tangan kanan.



Gambar 2.22 *Body frame*

dimana,  $i^b$ ,  $j^b$ , dan  $k^b$  adalah unit vektor dari bada UAV dimana badan UAV sebagai arah sumbu  $x, y$  dan  $z$ .

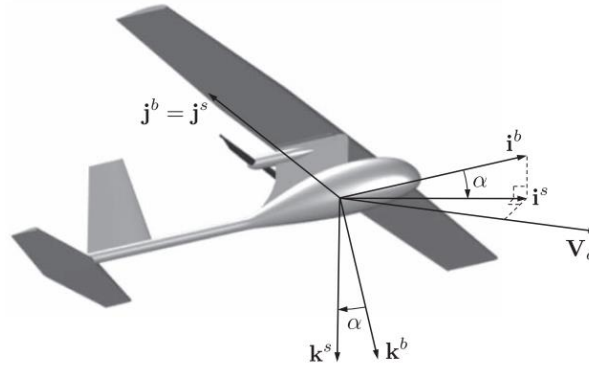
Apabila semua bentuk transformasi pada *vehicle frame* dan *body frame* digabungkan maka akan menjadi persamaan berikut

$$\mathcal{R}_v^b(\phi, \theta, \psi) = \mathcal{R}_{v2}^b(\phi) \mathcal{R}_{v1}^{v2}(\theta) \mathcal{R}_v^{v1}(\psi) \quad (2.1)$$



$$\mathcal{R}_v^b(\varphi, \theta, \psi) = \begin{pmatrix} \cos \theta \cos \psi & \cos \theta \sin \psi & -\sin \theta \\ \sin \varphi \sin \theta \cos \psi - \cos \varphi \sin \psi & \sin \varphi \sin \theta \sin \psi + \cos \varphi \cos \psi & \sin \varphi \cos \theta \\ \cos \varphi \sin \theta \cos \psi + \sin \varphi \sin \psi & \cos \varphi \sin \theta \sin \psi - \sin \varphi \cos \psi & \cos \varphi \cos \theta \end{pmatrix}$$

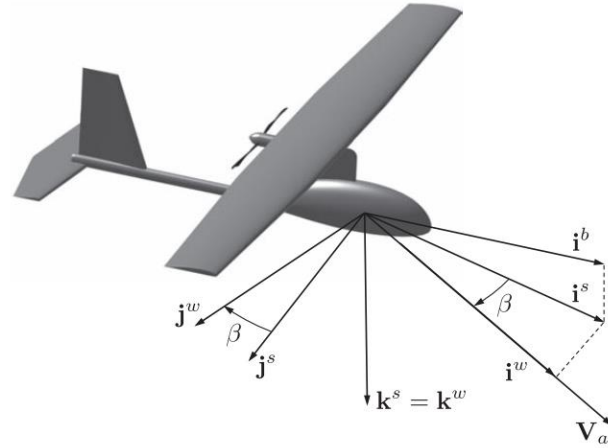
*Stability frame* ( $f^s$ ) adalah sudut yang dihasilkan antara rotasi badan pesawat pada sumbu  $y^b$  terhadap. Gaya aerodinamis terjadi saat gerakan dari UAV dipengaruhi oleh udara yang berada pada sekitar pesawat, dimana kecepatan udara dinotasikan dengan  $V_a$ . untuk menghasilkan gaya angkat (*lift*), sayap pada pesawat harus terbang pada sudut positif dengan tidak mengabaikan arah dari kecepatan udara. Sudut ini disebut dengan *angle of attack* yang dinotasikan dengan  $\alpha$  dimana pesawat akan berotasi pada sumbu  $y$  dari *body* pesawat.



Gambar 2.23 *Stability frame*

Sumbu  $J^b$  sama dengan sumbu sayap pesawat. Sedangkan sumbu  $i^s$  sesuai dengan sumbu dari arah kecepatan udara sedangkan  $i^b$  merupakan sumbu dari hidung pesawat, sehingga sudut  $\alpha$  merupakan sudut yang dihasilkan dari  $i^s$  dengan  $i^b$ .

*Wind frame* ( $f^w$ ) merupakan sudut antara arah angin dengan sumbu  $x^b$ - $z^b$  pada pesawat yang biasanya disebut dengan sudut side slip ( $\beta$ ).



Gambar 2.24 *Wind frame*

Sehingga total transformasi dari *body frame* terhadap *wind frame* adalah

$$\mathcal{R}_b^w(\alpha, \beta) = \mathcal{R}_s^w(\beta) \mathcal{R}_b^s(\alpha) \quad (2.2)$$

$$\mathcal{R}_b^w(\alpha, \beta) = \begin{pmatrix} \cos \beta \cos \alpha & \sin \beta & \cos \beta \sin \alpha \\ -\sin \beta \cos \alpha & \cos \beta & -\sin \beta \sin \alpha \\ -\sin \alpha & 0 & \cos \alpha \end{pmatrix} \quad (2.3)$$

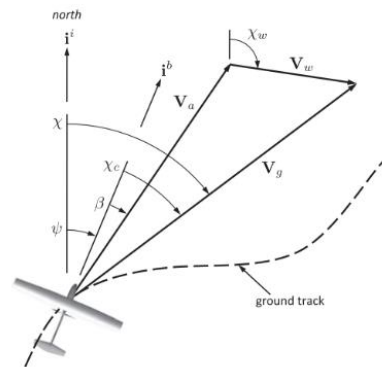
### **Wind Triangle**

*Wind triangle* menggambarkan hubungan navigasi pada pesawat antara *groundspeed vector*, arah kecepatan udara, dan arah angin. Sudut antara inertia North ( $x^i$ ) dan arah kecepatan inertia pada horizontal pesawat disebut dengan *course angle* ( $\chi$ ). *Crab angle* ( $\chi_c$ ) merupakan sudut antara *course angle* dan *heading* (arah hidung pesawat menurut kompas).

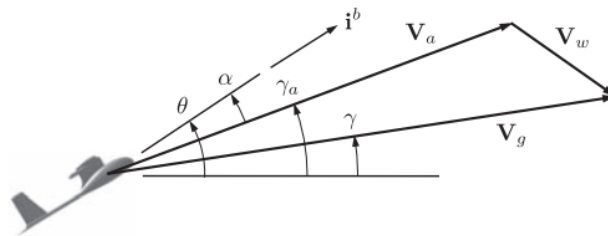
*Flight jalur angle* ( $\gamma$ ) merupakan sudut antara sumbu horizontal pesawat terhadap kecepatan pada bumi ( $V_g$ ). Sehingga terdapat dua sudut utama yang ditransformasikan dari *body frame* menuju *flight jalur frame* ( $\chi, \gamma$ ). Ada beberapa keadaan:

- *Crab angle* ( $\chi_c$ ) = 0
- *Sideslip angle* ( $\beta$ ) = 0
- $V_a = V_g$

Gambar 2.25 ini merupakan hubungan dari *wind triangle* secara horizontal sedangkan gambar 2.26 menampilkan hubungan dari *wind triangle* secara vertical.



Gambar 2.25 *Wind triangle* secara horizontal



Gambar 2.26 *Wind triangle* secara vertikal

### Parameter *Fixed Wing*

Parameter yang digunakan pada *paper* ini adalah parameter pada pesawat UltraStick-25e (Thor). Pada pesawat UltraStick-25e (Thor) memiliki *flap*, *aileron*, *rudder*, dan kontrol *elevator* yang biasa terdapat pada pesawat. Defleksi terbesar dari aktuator motor servo adalah  $25^\circ$ . Tabel 2.8 menampilkan parameter dari pesawat yang digunakan.

Tabel 2.8 Parameter *Fixed Wing*

<i>Property</i>	<i>Simbol</i>	<i>Nilai</i>	<i>Satuan</i>
<i>Wing span</i>	B	1.27	m
<i>Wing surface area</i>	S	0.3097	m <sup>2</sup>
<i>Maincord</i>	C	0.25	m
Massa	M	1.959	kg
Inersia	J <sub>x</sub>	0.07151	kg.m <sup>2</sup>
	J <sub>y</sub>	0.08636	kg.m <sup>2</sup>
	J <sub>z</sub>	0.15364	kg.m <sup>2</sup>
	J <sub>xz</sub>	0.014	kg.m <sup>2</sup>

### Kinematika *Fixed Wing*

- Koefisien Aerodinamika gerak Longitudinal

Gerak longitudinal adalah gerak  $x^b$ - $z^b$  terhadap badan pesawat yang dapat disebut dengan gerak *pitch*, dimana gerak tersebut akan dipengaruhi oleh gaya angkat (*lift force* ( $f_L$ ), *drag force* ( $f_D$ ) dan *pitch moment* ( $m$ ).

- Koefisien Aerodinamika gerak Lateral

Gerak lateral adalah gerak pada *yaw* dan *roll* pesawat. Gerak lateral dipengaruhi oleh *side force* ( $f_y$ ), momen *yaw* ( $r$ ), dan momen *roll* ( $p$ ).

### Dinamika *Fixed Wing*

Total gaya yang terjadi pada pesawat adalah:

Tabel 2.9 Parameter pada dinamika *Fixed Wing*

<b>Parameter</b>	<b>Keterangan</b>
$\alpha$	<i>Angle of attack</i>
$\beta$	<i>Side slip angel</i>
$J_x$	Inersia pada sumbu-x
$J_y$	Inersia pada sumbu-y
$J_z$	Inersia pada sumbu-z

---

$J_{xz}$	Inersia pada sumbu dibentang sumbu x dan z
$u, v, w$	Kecepatan inersia
$K_p$	Konstanta <i>thrust</i>
$K_d$	Konstanta drag
$m$	Massa total UAV
$g$	Grafitasi
$l_1$	Jarak antara motor 1 ke motor 5
$l_2$	Jarak antara motor 1 ke pusat massa UAV
$l_3$	Jarak antara motor 3 ke pusat massa UAV
$\Gamma$	Produk dari Matrix Inersia
$V_a$	Vektor <i>airspeed</i>
$V_g$	Vektor kecepatan <i>ground</i>
$V_w$	Vektor kecepatan angin
$C_L$	Koefisien angkat ( <i>fixed-wing</i> )
$C_D$	Koefisien drag ( <i>fixed-wing</i> )
$C_m.$	Koefisien momen <i>pitching</i>
$C_l.$	Koefisien momen <i>rolling</i>
$C_n.$	Koefisien momen yawing
$C_X.$	Koefisien gaya pada $X_B$
$C_Y.$	Koefisien gaya pada $Y_B$
$C_Z.$	Koefisien gaya pada $Z_B$
$k_{motor}$	konstanta efisiensi motor
$S_{prop}$	luas yang dihempas propeller
$C_{prop}$	Koefisien aerodinamis propeller
$S$	Area penampang sayap pesawat
$c$	<i>Maincord</i> sayap
$b$	Bentang sayap
$\rho$	Densitas atmosfir

---

### 2.2.2 Algoritma Theta\* [1]

Theta\* adalah algoritma pencarian *any-angle* yang berarti pencarian segala sudut. Secara empiris Theta\* dapat mencari jalan yang lebih pendek bila dibandingkan dengan algoritma A\*. Hal ini dikarenakan Theta\* menggabungkan kemampuan A\* pada grafik visibilitas (dimana perubahan arah terjadi hanya pada sudut-sudut dari sel yang terhalang) dan kemampuan A\* pada *grids* (dimana setiap sudut berkembang sejumlah dengan sel yang ada). Walaupun demikian, algoritma Theta\* memiliki kelemahan dalam proses pencarian yang lama. Perbedaan utama antara Theta\* dan A\* pada *grid* adalah *parent* dari sebuah *node* dapat berupa sudut manapun pada Theta\*, sedangkan *parent* dari sebuah *node* pada A\* harus berdekatan dengan *node* tersebut.

### 2.2.3 Metode State Dependent-Linier Quadratic [6]

Seluruh metode kontrol non-linier dengan penerapan rentang rendah menggunakan teknik linierisasi lokal untuk memodelkan sebuah *plant*. Meskipun Teknik ini menghasilkan permodelan yang sederhana, tetapi proses linierisasi yang terjadi harus sangat sering dilakukan agar sistem tidak meninggalkan wilayah linearisasi yang sangat kecil selama operasi berlangsung. Metode kontrol yang lebih maju seperti kontrol adaptif dan *gain scheduling* menggunakan linierisasi global yang lebih kompleks.

Pendekatan *State Dependent-Linier Quadratic* (SD-LQT) atau yang lebih dikenal *State Dependent Riccati Equation* (SDRE) menggunakan linierisasi semi-global untuk mengkrompomi permasalahan aplikasi dan kompleksitas yang terjadi. Kedua permasalahan tersebut ditangani oleh pendekatan dari metode SD-LQT karena metode ini berhubungan dengan metode *Linier Quadratic Tracking* (LQT)

### *Linear Quadratic Tracking* [5]

Konsep optimisasi sistem kontrol mengkompromikan pemilihan indeks performansi dan hasil desain dalam batas-batas kendali fisik. Perumusan indeks performansi berdasarkan persyaratan permasalahan, dan umumnya tidak hanya mencakup persyaratan performansi, tetapi juga batasan-batasan bentuk pengaturan yang menjamin keandalan fisik. Persyaratan yang harus dipenuhi biasanya disebut

sebagai spesifikasi performansi. Hal ini berkaitan dengan kestabilan mutlak, ketelitian, kestabilan relatif, dan kecepatan respon.

Di samping kestabilan mutlak, sistem kontrol harus memiliki kestabilan relatif yang layak. Jadi kecepatan respon harus cukup sistem kontrol harus memiliki kestabilan relatif yang layak. Jadi kecepatan respon harus cukup cepat dan menunjukkan peredaman yang layak. Suatu sistem kontrol harus dapat memperkecil kesalahan sampai nol atau batas yang dapat ditoleransi. Persyaratan kestabilan relatif yang layak dan ketelitian keadaan *steady state*, cenderung tidak dapat dipenuhi secara bersama-sama. Oleh karena itu, dalam mendesain sistem kontrol, dilakukan kompromi yang paling efektif diantara kedua persyaratan tersebut.

Indeks performansi adalah ukuran kuantitatif performansi dari sistem dan dipilih sehingga penekanan diberikan pada spesifikasi sistem yang dipentingkan, misalnya kesalahan dan sinyal aksi. Indeks performansi pasti berupa angka yang bernilai positif atau nol. Sistem kontrol terbaik didefinisikan sebagai sistem yang memiliki indeks performansi yang minimal. Desain kontrol optimal akan dilakukan melalui *variable state*. Pada dasarnya, solusi kontrol optimal dititik beratkan pada pencarian nilai sinyal kontrol yang optimal  $u(t)$  sehingga indeks performansi dioptimasi. *Linear Quadratic Tracking* merupakan salah satu metode kontrol optimal yang dikembangkan pada *plant* linier untuk mengatasi permasalahan *tracking*. Diketahui suatu *plant* dengan persamaan berikut:

$$\dot{x}(t) = A(t)x(t) + B(t)u(t) \quad (2.16)$$

$$y(t) = C(t)x(t) \quad (2.17)$$

Sinyal referensi dimodelkan dalam *state space*

$$\dot{x}(t) = Fz(t) \quad (2.18)$$

$$\tilde{y} = Hz(t) \quad (2.19)$$

Permasalahan dalam desain sistem kontrol dengan umpan balik *state* dalam menjaga *state* sistem pada nilai yang diinginkan. Apabila harga keluaran awal tidak nol. Permasalahannya adalah menentukan hukum kontrol sehingga vector kontrol memiliki kemampuan mengurangi kondisi awal sistem menuju nol secara cepat. Untuk mendapatkan sisten persamaan (2.16) dalam keadaan optimal, maka dipilih

vektor  $u(t)$  sedemikian rupa sehingga indeks performansi yang diberikan adalah minimum. Adapun indeks performansi kuadratik secara matematis dinyatakan sebagai berikut:

$$J = \int_0^\infty (e^T(t)Qe(t) + u^T(t)Ru(t))dt \quad (2.20)$$

Dimana  $e(t) = \tilde{y}(t) - y(t)$ , maka penjabaran indeks performansi sistem pada persamaan diatas adalah sebagai berikut:

$$\begin{aligned} J &= \int_0^\infty ((\tilde{y}(t) - y(t))^T Q (\tilde{y}(t) - y(t)) + u^T(t)Ru(t))dt \\ &= \int_0^\infty ((\tilde{y}(t)^T - y(t)^T)Q(\tilde{y}(t) - y(t)) + u^T(t)Ru(t))dt \\ &= \int_0^\infty ((y^T(t)Qy(t) - y^T(t)Q\tilde{y}(t) - \tilde{y}^T(t)Qy(t)) + \tilde{y}^T(t)Q\tilde{y}(t) + \\ &\quad u^T(t)Ru(t))dt \\ &= \int_0^\infty (x^T(t)C^TQCx(t) - x^T(t)C^TQHx(t) - z^T(t)H^TQCx(t) + \\ &\quad z^T(t)H^TQHx(t) + u^T(t)Ru(t))dt \\ &= \int_0^\infty \left( \begin{bmatrix} x(t) & z(t) \end{bmatrix} \begin{bmatrix} C^TQC & -C^TQH \\ -H^TQC & H^TQH \end{bmatrix} \begin{bmatrix} x(t) \\ z(t) \end{bmatrix} + u^T(t)Ru(t) \right) dt \end{aligned} \quad (2.21)$$

Sehingga didapatkan matriks Q dan R:

$$\tilde{Q} = \begin{bmatrix} C^TQC & -C^TQH \\ -H^TQC & H^TQH \end{bmatrix}, \quad \tilde{R} = R \quad (2.22)$$

Penentuan hukum kontrol optimal untuk sistem dengan indeks performansi tersebut memiliki arti praktis bahwa sistem mengkompromosikan kesalahan dan energi kontrol yang diminimumkan. Disamping meninjau kesalahan sebagai ukuran performansi sistem, juga memperhatikan energi yang diperlukan dalam aksi pengontrolan. Jika kesalahan diminimalkan tanpa memperhatikan energi yang diperlukan, maka suatu sistem kontrol akan memerlukan harga sinyal kontrol yang terlalu besar. Hal ini tidak diinginkan karena setiap sistem fisik akan mengalami saturasi. Saturasi pada aktuator dapat mengakibatkan kerusakan. Dengan demikian kondisi optimal sistem meminimumkan energi yang dinyatakan dengan bentuk kuadrat linier dari sinyal kontrol.

Kondisi optimal selanjutnya meminimumkan kesalahan dinyatakan dengan bentuk kuadratik *linier* dari *variable state*. Pendekatan SDRE dapat dikatakan sebagai pengembangan dari LQR. Metode ini menghasilkan kontroler non-linier sub-



optimal, seperti saat menggunakan persamaan Riccati untuk mengira-ngira solusi dari permasalahan kontrol optimal daripada menyelesaikan persamaan Hamilton Jacobi (*Hamilton Jacobi Equation/HJE*).

### Desain Kontroler SD-LQT/SDRE [6]

Salah satu keuntungan dari metode SDRE adalah adanya proses controller yang berkelanjutan secara sistematis. Hal ini yang harus diperhatikan sebagai kemiripan antara pendekatan SDRE dan LQT. SDRE memulai model dengan persamaan *input non-linier* sesuai dengan persamaan

$$\dot{x} = f(x) + g(x)u \quad (2.23)$$

Dengan persamaan indeks performansi

$$J = \frac{1}{2} \int_0^\infty x^T Q(x)x + u^T R(x)u dt \quad (2.24)$$

Dimana:

1.  $x \in R^n, u \in R^m$
2.  $f(x) \in C^k, g(x) \in C^k, Q(x) \in C^k, R(x) \in C^k, k \geq 1$
3.  $Q(x) = C^T(x)C(x) \geq 0$ , dan  $R(x) > 0$  untuk semua  $x$
4. Dan asumsikan bahwa  $f(0) = 0$  dan  $g(x) \neq 0$  untuk semua  $x$

Solusi dari masalah ini setara dengan penyelesaian menggunakan HJE. Namun, karena menyelesaikan permasalahan non-liier menggunakan HJE sulit dilakukan, oleh karena itu digunakanlah pendekatan SDRE/SD-LQT. Langkah-langkah perancangan sistem kontrol SDRE dapat dijelaskan sebagai berikut:

- a. Mengubah persamaan linier pada persamaan (2.23) menjadi bentuk *State Dependent Coefficient* (SDC). Bentuk SDC ini akan menghasilkan persamaan *state space* dengan matrik *state*  $A(x)$  dan  $B(x)$  yang *dependent* (masih mengandung *variable state*) sehingga nilai matriks akan terus berubah-ubah sesuai perubahan *state*. Selama simulasi berlangsung nilai matriks  $A(x)$  dan  $B(x)$  akan terus berubah hingga menghasilkan matriks yang konstan pada periode waktu tertentu.
- b. Menentukan matriks pembobot  $Q$  dan  $R$ . Pilihlah nilai pembobot  $Q$  dan  $R$  digunakan untuk menentukan performansi dari keseluruhan sistem yang

diharapkan dapat menghasilkan sinyal kontrol  $u$  yang optimal. Nilai pembobot  $Q$  dan  $R$  dapat merupakan nilai yang tetap atau berubah-ubah menggunakan metode *trial and error* atau berdasar perubahan *state*.

c. Menyelesaikan persamaan SDRE berikut:

$$A^T(x)P(x) + P(x)A(x) - P(x)B(x)R^{-1}(x)B^T(x)P(x) + C^T(x)Q(x)C(x) = 0$$

Penyelesaian ini digunakan untuk memperoleh matriks Riccati  $P(x)$  dengan bantuan matriks  $A(x)$ ,  $B(x)$  dan matriks pembobot  $Q$  dan  $R$ . Karena nilai matriks  $A(x)$  dan  $B(x)$  berubah-ubah, maka nilai  $P(x)$  juga mengikuti perubahan tersebut. Oleh karena itu matriks  $P(x)$  bukan merupakan konstanta melainkan berupa *variable* yang berubah tergantung pada *state* yang kemudian disebut bersifat *dependent* terhadap *state*. Persamaan inilah yang kemudian disebut persamaan Riccati yang *state dependent*.

d. Langkah selanjutnya menghitung nilai sinyal kontrol  $u$

$$u = -Kx$$

dengan

$$K = R^{-1}(x)B^T(x)P(x)$$

### **Parameterisasi *State Dependent Coefficient* (SDC) [8]**

Pada pendekatan SDRE, sinyal kontrol  $u$  juga merupakan sinyal umpan balik seperti LQR, tetapi umpan balik pada SDRE bergantung pada solusi dari SDRE itu sendiri. Metode ini disebut parameterisasi SDC, yaitu proses memfaktorkan sistem non linier menjadi semi linier dengan matriks *state* yang *dependent* seperti pada persamaan berikut:

$$\dot{x} = A(x)x + B(x)u$$

Dimana  $f(x) = A(x)x$  dan  $B(x) = g(x)$ . Persamaan ini dikenal sebagai persamaan *State Dependent Coefficient* (SDC). Perhatikan bahwa matriks  $A(x)$  dan  $B(x)$  merupakan fungsi *state* dari *plant*, dan kemudian menjadi koefisien dalam persamaan *Riccati* yang dijelaskan sebelumnya.

Perlu diingat bahwa parameter SDC tidaklah unik, tetapi memiliki kemungkinan mempunyai banyak matriks. Selain itu, penggunaan suatu parameter tertentu tidak menjamin suatu sistem dapat diselesaikan dengan menggunakan

pendekatan SDRE dengan asumsi bahwa *stabilizability* dan *detectability* mungkin dilanggar.

Meskipun teori dalam menentukan parameter telah ada, tetapi tidak menutup kemungkinan terjadi kesalahan dalam penentuan parameter. Menurut diskusi yang dilakukan oleh Cloutier membahas mengenai beberapa pendekatan untuk mendapatkan parameter optimal dari sejumlah parameter sub-optimal yang ada. Tetapi dalam penelitian ini akan focus pada contoh sistem dengan parameter SDC yang telah diketahui.

### ***Algebraic Riccati Equation (ARE) [5]***

*Algebraic Riccati Equation (ARE)* merupakan hal utama dalam metode LQR. Terdapat juga *Continuous-time Algebraic Riccati Equation (CARE)* yang dapat dituliskan pada persamaan (2.25).

$$A^T P + P B R^{-1} B^T + C^T Q C = 0 \quad (2.25)$$

Dimana  $A \in R^{n \times m}$ ,  $B \in R^{n \times m}$ ,  $Q \in R^{n \times n}$ , dan  $R \in R^{m \times m}$  konstan.

Jika persamaan (2.25) dibentuk menjadi matriks Hamilton, maka akan seperti persamaan (2.26).

$$M = \begin{bmatrix} A & -B R^{-1} B^T \\ -Q & -A^T \end{bmatrix} \quad (2.26)$$

Matriks Hamilton memiliki *eigen value* yang simetris terhadap sumbu imajiner. Untuk melihatnya perlu diketahui bahwa:

$$J^{-1} M J = -J M J = -M^T \quad (2.27)$$

dimana,

$$J = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \quad (2.28)$$

Dengan demikian, jika  $\lambda$  merupakan *eigen value* dari M, begitu juga dengan  $-\lambda$ . Matriks Hamilton dapat digunakan untuk perhitungan numerik dari penyelesaian CARE. CARE memiliki beberapa solusi karena merupakan sebuah matriks persamaan kuadrat yang simetris terhadap matriks P.

### Pemilihan Matriks Q dan R [7]

Matriks pembobot Q dan R menentukan hasil dari persamaan Riccati (2.25). Desain parameter untuk matriks pembobot dapat berpengaruh pada keseluruhan performansi sistem. Pemilihan harga matriks Q dan R berfungsi untuk meminimumkan IP. Meskipun tidak ada metode khusus untuk menentukan kedua matriks pembobot ini, terdapat beberapa prosedur untuk memilih nilai matriks pembobot Q dan R yang tepat.

Matriks Q merupakan koefisien pembobot yang digunakan untuk menentukan lebar area *state*, sedangkan matriks R digunakan untuk menentukan lebar sinyal kontrol *u*. Secara umum penentuan matriks pembobot Q dan R berpedoman pada:

1. Semakin besar harga matriks Q maka akan memperbesar harga elemen matriks sinyal kontrol dan mempercepat sistem mencapai *steady state*.
2. Semakin besar harga matriks R, maka akan memperkecil harga elemen matriks sinyal kontrol dan memperlambat sistem mencapai *steady state*.

Tetapi perlu diperhatikan bahwa apabila sinyal kontrol terlalu besar dapat menyebabkan saturasi pada peralatan actuator. Oleh karena itu, kedua matriks ini harus ditentukan agar *state* dan *input* dibatasi sehingga tidak terlalu besar (berdasarkan *physical constraints*, dll). Beberapa prosedur yang dapat digunakan dalam pemilihan matriks Q dan R adalah [8].

- a) Pilih matriks R berupa matriks diagonal untuk mempermudah perhitungan. Pemberian matriks pembobot Q dan R dalam bentuk diagonal akan mempermudah perhitungan karena dengan demikian akan didapatkan batas tiap variable yang independen terhadap satu sama lain.
- b) Jika tidak ingin memberikan nilai yang terlalu tinggi pada variable *state* atau kontrol *input*, maka matriks pembobot harus memiliki nilai yang tinggi.
- c) Pilih matriks Q dan R diagonal. Untuk mempercepat respon, pilih seluruh factor konstan, tetapi akan membutuhkan kontrol yang lebih tinggi.
- d) Matriks Q semi definit dan R harus matriks definit positif.

## **BAB 3**

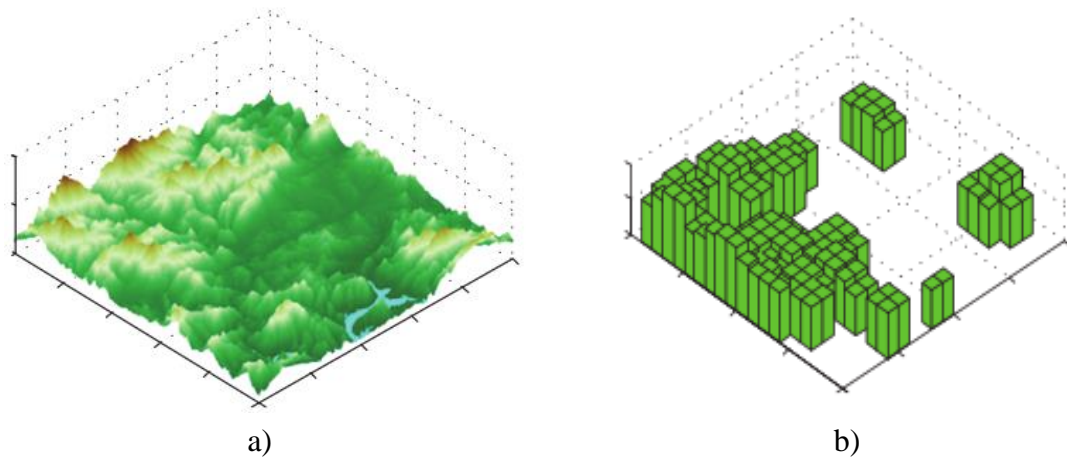
### **METODOLOGI PENELITIAN**

Pada bab ini akan membahas mengenai langkah kerja penelitian yang akan dilakukan dalam perancangan algoritma jalur *planning* yang dapat dilalui oleh *Fixed Wing UAV* pada *urban environment* yang akan direpresentasikan sebagai blok-blok persegi pada sebuah *roadmap* yang akan dibuat. Diberikan titik awal dan titik target pada *roadmap* yang sudah dilengkapi dengan *nodes* dan *edge*, lalu algoritma pencarian jalur digunakan untuk mendapatkan jalur terpendek pada *nodes* yang telah disediakan. Jalur yang didapatkan oleh algoritma pencarian perlu dihaluskan karena jalur yang dihasilkan dapat berupa pembelokan yang tajam, oleh karena itu akan digunakan sebuah metode penghalusan jalur dengan mempertimbangkan kemampuan *maneuver* dari *Fixed Wing UAV*, dan untuk memastikan jalur yang sudah dihaluskan dapat dilalui, maka *Fixed Wing UAV* akan dibuat untuk mengikuti jalur tersebut. *Fixed Wing UAV* yang akan digunakan pada penelitian ini adalah *Fixed Wing ultrastick-25*.

### **3.1 Memodelkan Lingkungan 3D**

#### **3.1.1 Memodelkan Digital Map dan Convex Obstacle**

Pembuatan obstacle 3D yang merepresentasikan *urban environment* dimodelkan dengan menggunakan data elevation map dengan satu set *convex obstacle* dengan menghitung *convex hull* berdasarkan sampel *map digital* dalam  $1\text{km}^2$  yang diberikan. Gambar (3.1.b) menunjukkan contoh *generate* pemodelan *convex obstacle* berdasarkan peta yang diberikan dari gambar (3.1.a) yang diolah menggunakan DTED standar sehingga menghasilkan blok-blok persegi yang merepresentasikan *urban environment* yang disebut sebagai *map M*.



Gambar 3.1 a) Contoh *Digital Map*, b) *Convex Obstacle 3D*

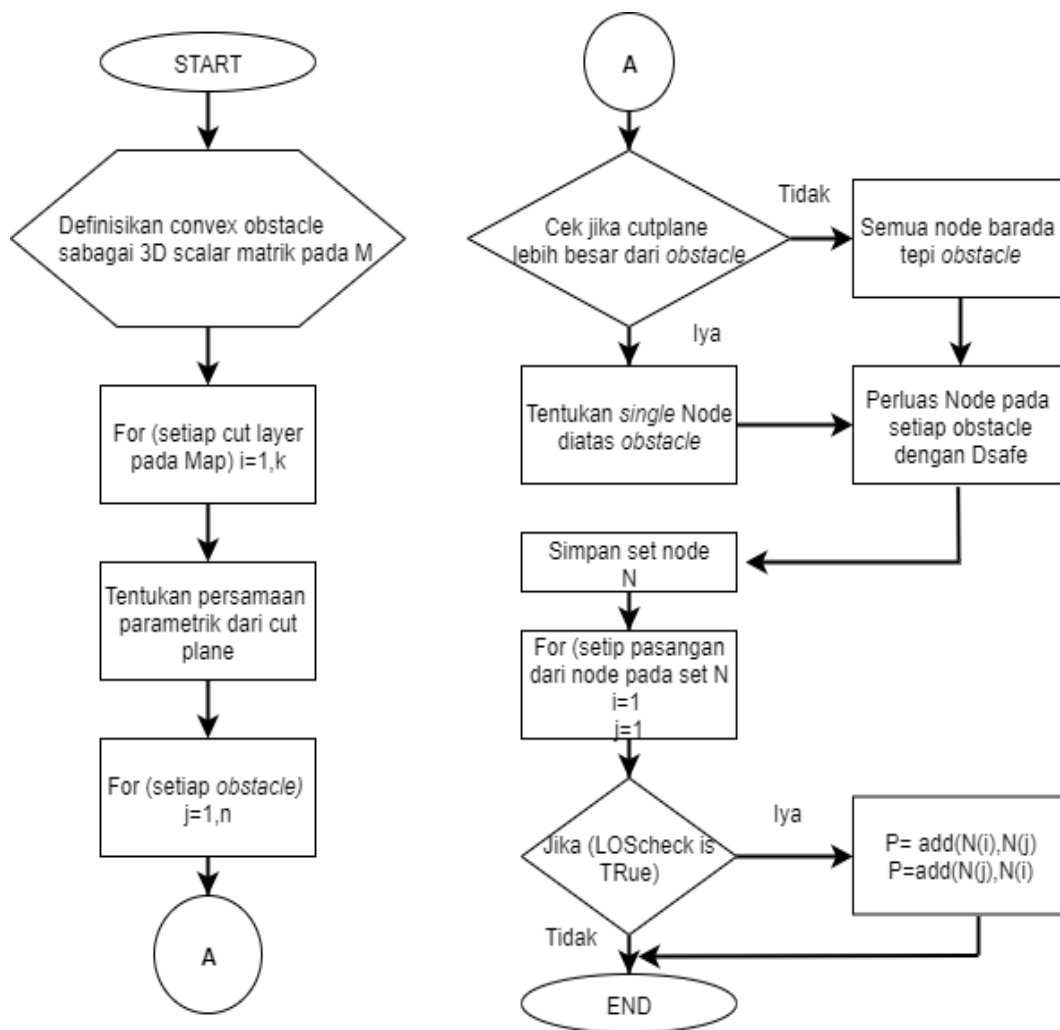
### 3.1.2 Algoritma pembuatan Roadmap

Dalam subbab ini, penulis akan menyajikan deskripsi peta jalan dan algoritma pembuatannya. *Digital Map* ( $M$ ) yang dipresentasikan oleh nilai ketinggian dalam sebuah matriks skalar 3D. Roadmap  $\Gamma$  adalah seperangkat *node* (*node*;  $N$ ) dan koneksi jalur (*edge*;  $P$ ) antara *node*. Secara matematis, ini adalah grafik tepi ganda terarah di mana semua *nodes* yang terlihat saling berhubungan.

Untuk membangun *roadmap*, langkah pertama melibatkan pengambilan sampel 3D untuk *node* yang memungkinkan. *Map* ini dibagi menjadi  $k$ -layer oleh bidang horizontal  $k$  dengan jarak potong yang sama dari  $H_{min}$  ke  $H_{max}$ . Membuat *node* di dalam batasan *obstacle* yaitu dengan menghitung titik-titik persimpangan bidang potong dan tepi *obstacle*. Jika bidang potong lebih tinggi dari *obstacle*, maka kita tentukan *single node* yang terletak di atas *obstacle* yang memiliki ketinggian yang sama dengan bidang potong. Setelah itu, kita memperluas pembuatan *node* pada *obstacle* dengan jarak yang aman. Koordinat *nodes* yang diperluas harus disimpan dalam set  $N$  (*node*). Setelah menemukan set *node*  $N$ , selanjutnya membuat set *edge*  $P$  dengan pemeriksaan visibilitas. Kompleksitas waktu dari algoritma sangat tergantung pada fungsi pemeriksaan *line of sight*. Karena karakteristik *convex obstacle*, kita dapat memperoleh metode untuk menghitung *line of sight* dalam waktu linier. Ada dua kasus yang berbeda dalam prosedur pemeriksaan; misalnya, pasangan *node* berada dalam *obstacle* yang sama atau terletak pada *obstacle* yang berbeda. Karena *obstacle* memiliki karakteristik *convex*, *node* awal

dan tujuan memiliki *line of sight* jika mereka berada di sisi yang sama. Kalau tidak, mereka tidak saling bertemu. Pada kasus kedua *node* berasal dari obstacle yang berbeda, kami memproses dengan mengeksplorasi algoritma persimpangan cembung untuk memeriksa visibilitas.

Secara umum perancangan algoritma *roadmap* dapat digambarkan dalam bentuk *flowchart* berikut ini.



Gambar 3.2 *Flowchart* perancangan algoritma *roadmap*

### 3.2 Theta\* dan Pencarian Jalur Terpendek

Sebelum membangun algoritma, notasikan bahwa  $S$  adalah semua *node* atau *node* yang terbentuk dalam peta 3D, dimana  $s_{start} \in S$  adalah *node* awal dari pencarian, dan  $s_{goal} \in S$  adalah *node* tujuan dari pencarian.  $succ(s) \subseteq S$  adalah set *node* tetangga dari  $s \in S$  yang memiliki *line-of-sight* pada  $s$ .  $c(s, s')$  adalah jarak *straight-line* antara  $s$  dan  $s'$ , dan *line-of-sight* ( $s, s'$ ) adalah benar jika dan hanya jika mereka memiliki *line-of-sight*.

Dalam perancangan algoritma juga memelihara dua struktur data global yaitu: (1) *Open List*, antrian prioritas yang berisi *node-node* yang harus dipertimbangkan untuk ekspansi. (2) *Close List*, yang berisi *node-node* yang sudah di perluas dan dipastikan bahwa masing-masing *node* sudah diperluas satu kali. Theta\* mengevaluasi *node* dengan menggabungkan  $g(n)$ , yaitu *cost* untuk mencapai *node*, dan  $h(n)$ , yaitu *cost* yang diperlukan dari *node* untuk mencapai tujuan, dalam notasi matematika dituliskan sebagai persamaan (3.1):

$$f(n) = g(n) + h(n) \quad (3.1)$$

dimana:

$f(n)$  = Biaya evaluasi

$g(n)$  = Biaya yang sudah dikeluarkan dari keadaan awal sampai keadaan  $n$

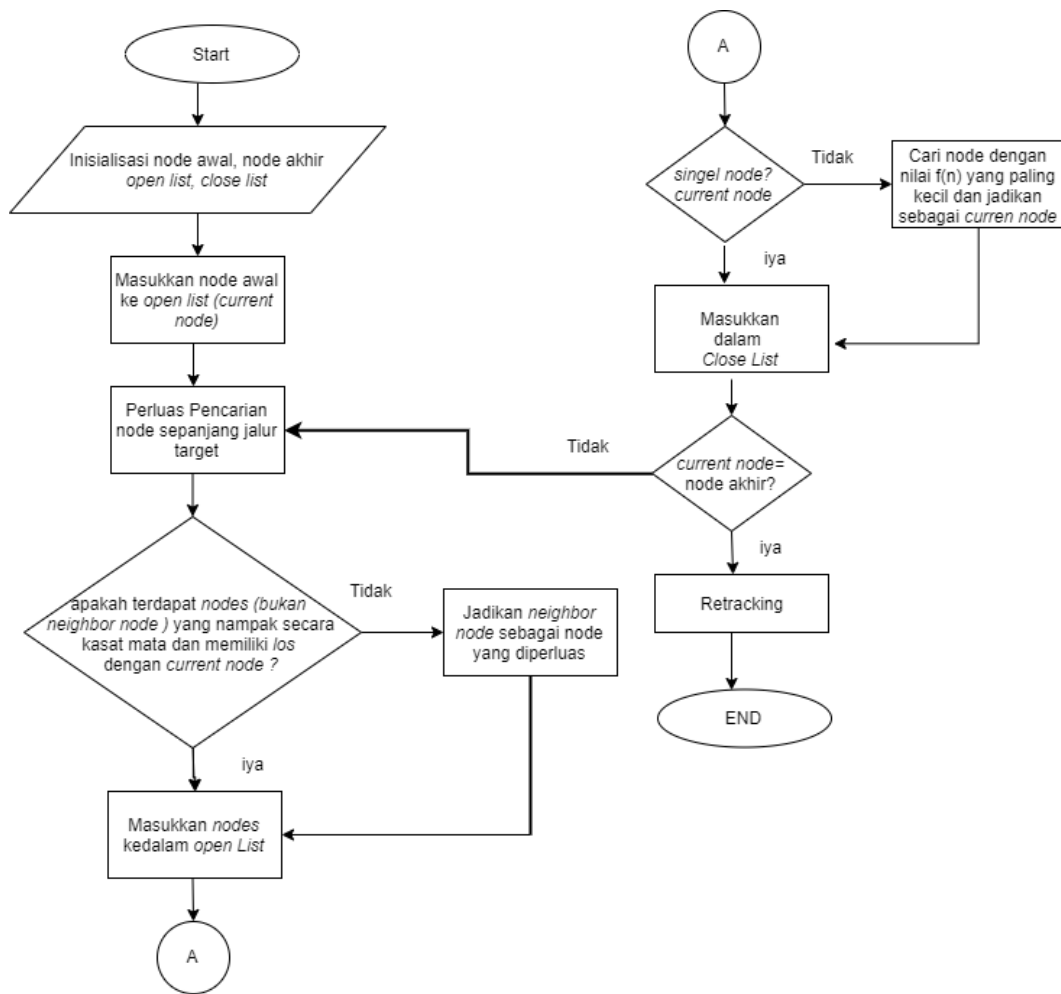
$h(n)$  = Estimasi biaya untuk sampai pada suatu tujuan mulai dari  $n$

Biaya dari fungsi diatas dapat menggunakan metode *Euclidean distance* untuk menentukan jarak antara 2 *node*. *Euclidean distance* ini berkaitan dengan Teorema Pythagoras dan biasanya diterapkan pada 1, 2 dan 3 dimensi. Pada penelitian ini posisi *node* berada dalam ruang 3D sehingga rumus *Euclidean distance* menjadi seperti pada persamaan (3.2).

$$j = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2} \quad (3.2)$$

Secara garis besar rancangan pembuatan algoritma Theta\* sebagai pencari jalur terpendek pada *urban environment* yang sudah disajikan dapat dirangkum dalam bentuk sebuah flowchart berikut.





Gambar 3.3 Flowchart algoritma Theta\*

### 3.3 Fixed Wing Ultrastick-25 (Manuever)

Pada penelitian ini *Fixed Wing* ultrastick-25 UAV digunakan sebagai *plant* yang akan melintasi jalur terpendek yang sudah didapatkan oleh algoritma Theta\*. Algoritma Theta \* tidak menganggap kinematika kendaraan sebagai bagian dari pembuatan jalur oleh karena itu permasalahan ini menjadi masalah utama untuk *nonholonomic vehicles* seperti *Fixed Wing* UAV, yang membutuhkan proses perataan untuk merealokasi urutan *node* untuk mendapatkan jalur yang dapat diterbangkan. Solusi yang akan digunakan pada peneitian ini adalah menggunakan metode *Bezier curve* sebagai *smoothing* jalur.

*Fixed Wing* UAV merupakan UAV yang memiliki kemampuan belok yang terbatas oleh karena itu pada perancangan *Bezier curve* akan mempertimbangkan kemampuan *maneuver* dari *Fixed Wing* ultrastick-25 UAV sehingga jalur yang dihasilkan oleh *Bezier curve* merupakan jalur yang dapat dilewati oleh *Fixed Wing*. Kemampuan *maneuver* dari *Fixed Wing* ultrastick-25 UAV akan didapatkan dengan mensimulasikan persamaan dari model *Fixed Wing* ultrastick-25 (3.3) - (3.12)

$$\begin{aligned} \dot{p}_n = & (\cos \theta \cos \psi)u + (\sin \phi \sin \theta \cos \psi - \cos \phi \sin \psi)v + (\cos \phi \sin \theta \cos \psi \\ & + \sin \phi \sin \psi)w \end{aligned} \quad (3.3)$$

$$\begin{aligned} \dot{p}_e = & (\cos \theta \sin \psi)u + (\sin \phi \sin \theta \sin \psi - \cos \phi \sin \psi)v + (\cos \phi \sin \theta \sin \psi \\ & + \sin \phi \cos \psi)w \end{aligned} \quad (3.4)$$

$$\dot{h} = u \sin \theta - v \sin \phi \cos \theta - w \cos \phi \cos \theta \quad (3.5)$$

$$\begin{aligned} \dot{u} = & rv - qw - g \sin \theta + \frac{\rho V_a^2 S}{2m} \left[ C_x(\alpha) \frac{cq}{2V_a} + C_{x_{\delta e}}(\alpha) \delta_e \right] \\ & + \frac{\rho S_{prop} C_{prop}}{2m} \left[ (k_{moto} \delta_t)^2 - V_a^2 \right] \end{aligned} \quad (3.6)$$

$$\begin{aligned} \dot{v} = & pw - ru - g \cos \theta \sin \phi + \frac{\rho V_a^2 S}{2m} \left[ C_{Y_0} + C_{Y_\beta} \tan^{-1} \left( \frac{v}{\sqrt{u^2 + w^2}} \right) + C_{Y_p} p \right. \\ & \left. + C_{Y_r} r \frac{br}{2V_a} + C_{Y_{\delta a}} \delta_a + C_{Y_{\delta r}} \delta_r \right] \end{aligned} \quad (3.7)$$

$$\dot{w} = qu - pv - g \cos \theta \cos \phi + \frac{\rho V_a^2 S}{2m} \left[ C_z(\alpha) + C_{z_q}(\alpha) \frac{cq}{2V_a} + C_{z_{\delta e}}(\alpha) \delta_e \right] \quad (3.8)$$

$$\begin{aligned} \dot{p} = & \Gamma_1 pq - \Gamma_2 qr + \frac{\rho V_a^2 S b}{2m} \left[ C_{p_0} + C_{p_\beta} \tan^{-1} \left( \frac{v}{\sqrt{u^2 + w^2}} \right) + C_{p_p} \frac{bp}{2V_a} \right. \\ & \left. + C_{p_r} \frac{br}{2V_a} + C_{p_{\delta a}} \delta_a + C_{p_{\delta r}} \delta_r \right] \end{aligned} \quad (3.9)$$

$$\dot{q} = \Gamma_5 pr - \Gamma_6 (p^2 - r^2) + \frac{\rho V_a^2 S c}{2J_y} \left[ C_{m_0} + C_{m_\alpha} + C_{m_q} \frac{cq}{2V_a} + C_{m_{\delta e}} \delta_e \right] \quad (3.10)$$

$$\dot{r} = \Gamma_7 pq - \Gamma_1 qr + \frac{\rho V_a^2 S b}{2} \left[ C_{r_0} + C_{r_\beta} \beta + C_{r_p} \frac{bp}{2V_a} + C_{r_r} \frac{br}{2V_a} + C_{r_{\delta a}} \delta_a + C_{r_{\delta r}} \delta_r \right] \quad (3.11)$$

$$\dot{\phi} = p + q \sin \phi \tan \theta + r \cos \phi \tan \theta \quad (3.12)$$

$$\dot{\psi} = q \sin \phi \sec \theta + r \cos \phi \sec \theta \quad (3.13)$$

$$\dot{\theta} = q \cos \phi - r \sin \phi \quad (3.14)$$

### 3.4 Strategi *Smoothing* menggunakan *Bezier curve*

Permasalahan *smoothing* jalur dirumuskan sebagai masalah optimisasi dengan *constraint*. Tujuannya adalah untuk menghitung jalur yang diperkirakan pada *real* jalur yang memungkinkan UAV membuat *maneuver* yang paling sedikit. Perhatikan bahwa ada hubungan antara kelayakan jalur dan optimalitasnya. Dengan *smoothing*, optimalitas jalur dalam hal keselamatan dan panjang jalur dapat menurun. Gagasan yang diusulkan di sini adalah untuk menetapkan koridor keselamatan di sekitar *real* jalur sebagai berikut:

$$1. (Q_j - p(t_j)) \leq (D_j - \delta), \quad (3.15)$$

Dimana:

$Q_j$  = Data asli

$p(t_j)$  = Nilai data yang diperkirakan dihitung oleh interpolasi

$D_j$  = Jarak *Euclidean* ke *obstacle* terdekat di setiap titik jalur asli.

2. Koefisien kelengkungan kurva

$$a. \frac{d^2 y}{dx^2} = \frac{d^2 y / dt^2}{(dx / dt)^2} < \frac{1}{R^2} \quad (3.16)$$

$$b. \frac{d^2 z}{dp^2} = \frac{d^2 z / dt^2}{(dx / dt)(dy / dt)} < \frac{1}{R^2} \quad (3.17)$$

Dimana:

$R^2$  = Manuver minimum *Fixed Wing*

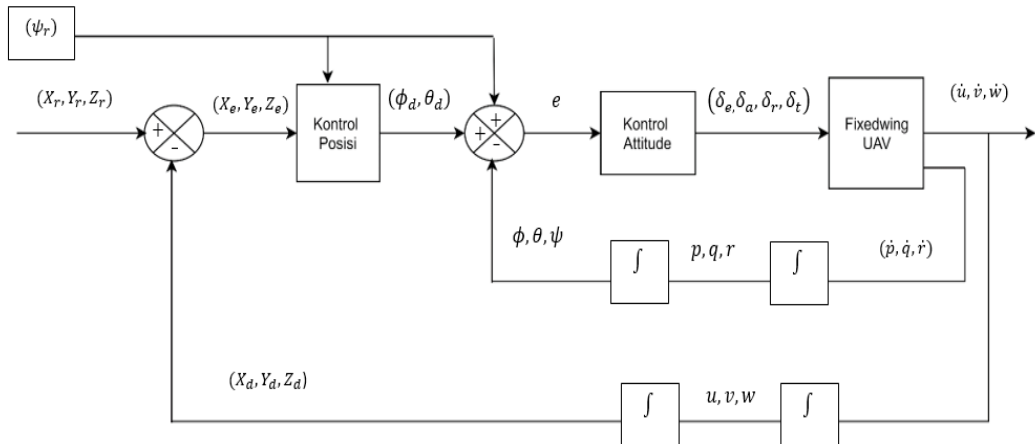
Jalur yang mulus harus berada di koridor keselamatan. Untuk itu, kita harus menentukan (dengan memecahkan masalah optimisasi) titik kontrol yang menentukan bentuk jalur yang mulus yang dibentuk oleh *patches Bezier curve* yang terhubung. Properti interpolasi titik akhir dari *Bezier curve* memastikan bahwa lintasan dimulai pada  $P_s$  dan berakhir pada  $P_T$ . Fungsi objektif yang akan diminimalkan adalah rata-rata jarak kuadrat antara *real* data dengan data yang dihaluskan.

Misalkan ada  $m$  node  $Q_j$  ( $j = 1: m$ ) dalam data asli. Untuk data yang dihaluskan  $p(t_j)$ , diketahui bahwa  $0 \leq t_j \leq 1$  dengan pembagian yang sama, maka  $Q_1 = p(0) = P_s$  dan  $Q_m = p(1) = P_T$ . Maka fungsi objektifnya menjadi:

$$\min_{p(t_j)} \left( \frac{1}{m-2} \sqrt{\sum_{j=2}^{m-1} (Q_j - p(t_j))^2} \right) \quad (3.18)$$

### 3.5 Perancangan Kontrol *Fixed Wing*

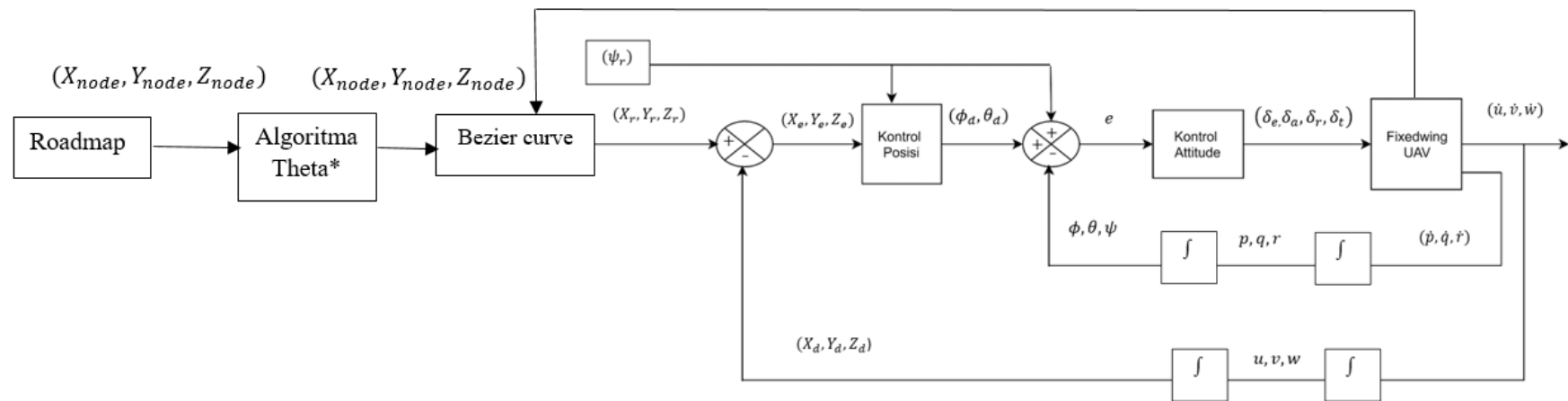
Skema kontrol *Fixed Wing* dibagi menjadi kontroler *attitude*, dan kontroler posisi. Kontroler yang akan digunakan pada penelitian ini adalah *State Dependent-Linear Quadratic Control* (SD-LQT) dimana kontroler ini mengatur gerak *Fixed Wing* pada jalur yang sudah didapatkan sebelumnya. Dalam kontrol *Fixed Wing* terdapat *inner loop* dan *outer loop* dimana *inner loop* sebagai kontrol *attitude* dan *outer loop* sebagai kontrol posisi. Masukan dari kontrol posisi yaitu  $(X_e, Y_e, Z_e)$  dan keluaran nya berupa  $(\phi_d, \theta_d)$  dan masukan heading  $(\psi_d)$  yang akan menjadi masukan untuk kontrol *attitude* seperti yang dijelaskan pada gambar (3.4)



Gambar 3.4 Skema kontrol *Fixed Wing*

### 3.6 Diagram Blok Keseluruhan

Pada tahapan perancangan diatas dapat diketahui tahapan-tahapan yang akan dikerjakan pada penelitian ini. Mulai dari perancangan peta 3D *urban environment*, pencarian jalur terpendek, *smoothing* jalur, serta pengujian jalur dengan membuat sistem *Fixed Wing* melalui jalur tersebut. Perancangan di atas juga dapat dilihat dalam bentuk blok diagram sistem sebagai berikut.



Gambar 3.5 Blok Diagram Sistem Keseluruhan

### 3.7 Hipotesa Penelitian

Berdasarkan kajian pustaka yang telah dilakukan dan rancangan konseptual yang telah dibuat, penulis memiliki beberapa hipotesa penelitian antara lain:

1. Pencarian jalur pada *urban environment* menggunakan algoritma Theta\* diharapkan dapat menemukan jalur terpendek dengan waktu tercepat.
2. Dengan menggunakan metode *Bezier curve* (dengan syarat manuver) untuk penghalusan jalur diharapkan dapat menghaluskan jalur yang tajam sehingga jalur dapat dilewati oleh *Fixed Wing UAV*.
3. Pada saat *tracking* jalur, metode kontrol yang digunakan dapat menghasilkan waktu minimum dan kesalahan yang kecil dalam mencapai target

### 3.8 Rencana Pengujian

Pengujian akan dilakukan untuk membuktikan hipotesa yang telah dibuat. Rencana pengujian yang akan dibuat adalah sebagai berikut:

1. Menguji dengan *start point* dan *target point* yang berbeda-beda pada banyak variasi bangunan/*obstacle*.
2. Menguji dengan mengikuti (*Tracking*) path yang sudah dihaluskan menggunakan *Bezier curve* (tanpa syarat manuver)
3. Menguji dengan mengikuti (*Tracking*) path yang sudah dihaluskan menggunakan *Bezier curve* (dengan syarat manuver)

### 3.9 Kriteria Pengujian

Kriteria pengujian yang akan dicapai adalah:

1. Jalur yang dihasilkan merupakan jalur terpendek dengan waktu tercepat dengan membandingkan hasilnya dengan kajian pustaka 2.
2. Hasil *tracking* pada jalur yang telah dihaluskan dengan *Bezier curve* (dengan manuver) menghasilkan *error* yang lebih kecil daripada hasil *tracking* jalur dengan *Bezier curve* (tanpa manuver).

## **BAB 4**

### **RENCANA DAN JADWAL KEGIATAN**

Pada bab 4, membahas tentang rencana dan jadwal kegiatan dari tesis yang akan dikerjakan oleh penulis.

#### **4.1 Rencana Kegiatan**

Beberapa hal yang dilakukan dalam rencana kegiatan penelitian ini adalah sebagai berikut:

1. Kajian Pustaka

Tahap awal yang dilakukan adalah kajian pustaka untuk mendapatkan permasalahan, ide, tren saat ini, dasar teori dan banyak hal lain yang dibutuhkan untuk penelitian. Selanjutnya, merumuskan permasalahan yang diangkat pada tesis, menyusun rencana penyelesaian dan menyusun rencana pengujian. Dengan kajian pustaka, penulis diharapkan dapat menghasilkan karya tulis dengan pembaharuan yang lebih baik.

2. Perancangan Sistem

Tahap perancangan sistem terdiri dari perancangan algoritma pembuatan *Map*, pencarian jalur terpendek serta memuluskan jalur yang sudah didapatkan. Rancangan akan dibuat menggunakan fasilitas *script* M-File dan *Simulink* pada MATLAB R2018b.

3. Pengujian Sistem

Pengujian sistem dilakukan setelah perancangan sistem secara keseluruhan selesai dilakukan, sehingga dapat dilakukan pengujian pada masing-masing perancangan.

4. Analisa dan Tinjau Ulang

Setelah perancangan dan pengujian sistem selesai dilakukan sehingga diperoleh hasil pengujian, maka selanjutnya hasil tersebut akan dianalisa. Apabila hasil analisa dari hasil pengujian telah mampu menyelesaikan masalah dan menjawab tujuan dari penelitian, maka tahap selanjutnya adalah membuat kesimpulan dari penelitian yang telah dilakukan. Namun, apabila hasil analisa belum dapat menyelesaikan permasalahan dan belum memenuhi tujuan yang

ingin dicapai, maka akan dilakukan perbaikan agar dapat menyelesaikan masalah dan tujuan penelitian.

#### 5. Penyusunan Laporan

Tahap ini dapat dilakukan setelah hasil dari pengujian telah menyelesaikan permasalahan yang telah diangkat pada tesis ini. Dalam penyusunan laporan harus memuat semua kegiatan penelitian dalam bentuk laporan, baik hasil penelitian, Analisa maupun kesimpulan. Pada tahap ini juga dilakukan penyusunan *paper* yang akan dipublikasikan.

#### 6. Konsultasi dengan Dosen Pembimbing

Pada tahap ini, konsultasi dilakukan untuk memperoleh masukan, koreksi dan pemecahan masalah yang terjadi pada saat penelitian.

### 4.2 Jadwal Kegiatan

Jadwal kegiatan penelitian yang akan dilakukan kurang lebih selama sembilan bulan ditunjukkan pada Tabel 4.1.

Tabel 4.1 Jadwal Kegiatan

No	Kegiatan	Bulan								Target
		8	9	10	11	12	1	2	3	
1	Kajian pustaka									Memahami metode dan dasar teori yang digunakan.
2	Perancangan sistem									Merancang sistem pada MATLAB berdasarkan pada metode dan dasar teori yang digunakan.
3	Pengujian sistem									Mengetahui kemampuan dari sistem yang telah dirancang.



4	Analisa dan tinjau ulang								Apakah sistem yang dirancang sudah sesuai dengan tujuan yang ingin dicapai atau belum.
5	Penyusunan laporan								Tesis dan <i>paper</i> .
6	Konsultasi dengan dosen pembimbing								Mendapatkan masukan, koreksi, pemecahan masalah yang diteliti dan penyelesaian karya tulis.

*Halaman ini sengaja dikosongkan*

## DAFTAR PUSTAKA

- [1] Luca De Filippis ,Giorgio Guglieri, and Fulvia Quagliotti,” *Jalur Planning Strategies for UAVs in 3D Environments,*" *Hindawi,Journal of intelligent & Robotic System*, vol. 65,no.1-4 pp. 247-264 2012
- [2] Zahoor Ahmad, Farman Ullah,Chong Tran,dan Sungchan lee," *Efficient Energy Flight Jalur Planning Algorithm Using 3-D Visibility Roadmap for Small Unmanned Aerial Vehicle,*" *Hindawi,International Journal of Aerospace Engineering*,2017.
- [3] Karima Benzaid, Romai Marie, Noura Mansouri, dan Ouidad Labbani-Igbida, “*Filtered Medial Surface Based Approach for 3D Collision-Free Jalur Planning Problem,*” *Hindawi,International Journal of Aerospace Engineering*, 2018
- [4] R. W. Beard and T. W. McLain, *small unmanned Aircraft Theory and Practice*, Oxford: Princeton University Press, 2012.
- [5] Frank L.Lewis and Vassilis L.syrmos, *Optimal Control*, John Wiley and son, 1995
- [6] Rahma Nur Amalia,"Perancangan Kontroller State Dependent LQT untuk Jalur *Following* dan Jalur *Tracking* pada *Autonomous Underwater Vehicle* (AUV),Thesis ,Jurusan Teknik Elektro ITS,2015.
- [7] Katsev Sergey,Streamlining of the State-Dependent Riccati Equation Controller Algorithm for an Embedded Implementtion,"*Thesis*,Departemen of Computer Enginerring,2016.
- [8] Utami, Dyah Tri, "Perancangan Kontroler State Dependent Riccati Equation untuk Stabilisasi Pendulum Terbalik Dua Tingkt", Tugas Akhir, Jurusan Teknik Elektro ITS, 2010.

