BAB 3
ANALISIS DAN PERANCANGAN

3.1. Analisis Masalah

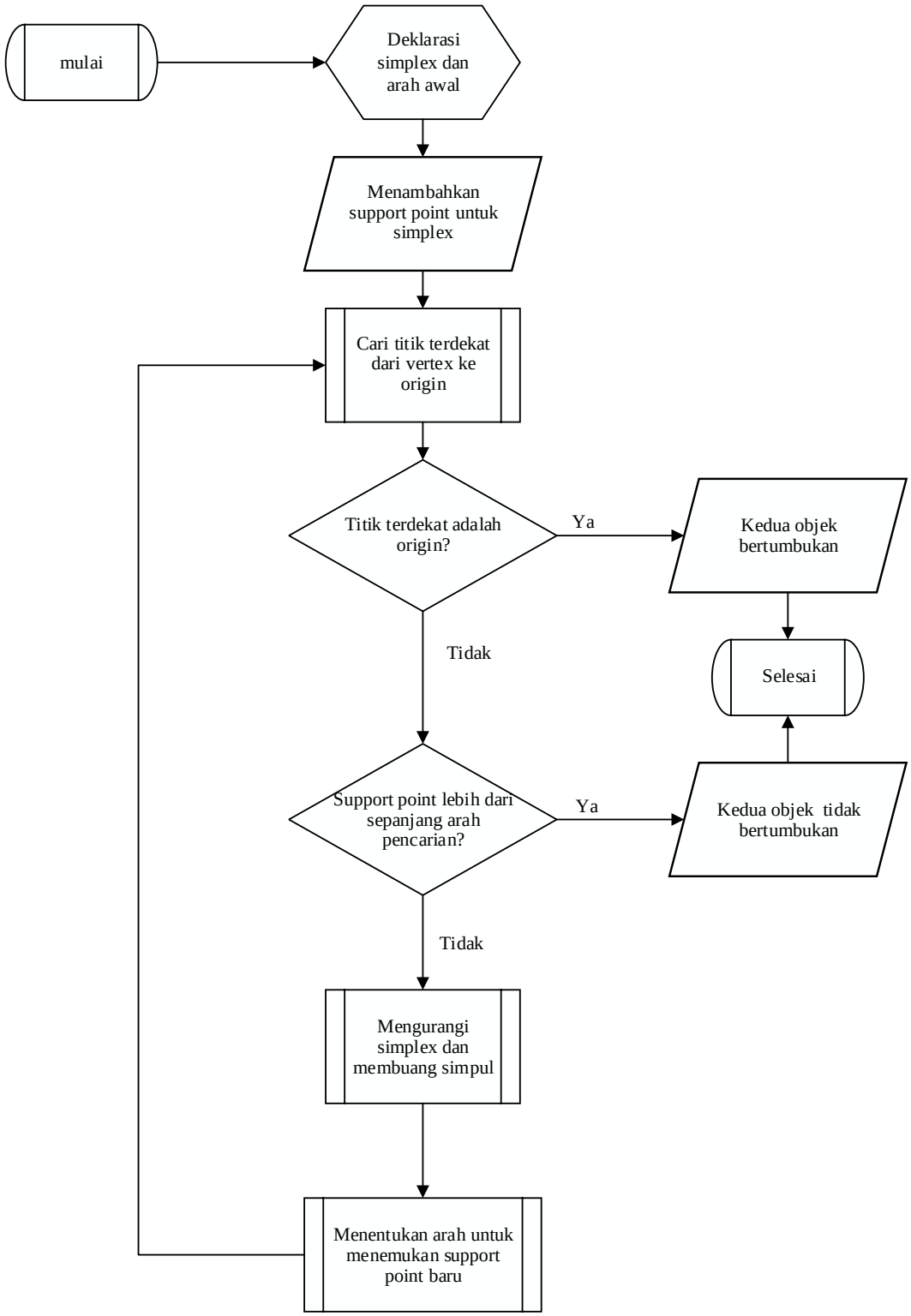
Analisis masalah pada pembangunan aplikasi pembelajaran fisika momentum dan impuls yaitu sebagai berikut :

1. Bagaimana cara mensimulasikan momentum dan impuls untuk menambah fasilitas pendukung pembelajaran mata pelajaran fisika?
2. Bagaimana memodelkan simulasi fisika momentum dan impuls dengan *impact* yang sesuai dengan parameter masukan bola?
3. Bagaimana mengimplementasikan algoritma gilbert jhonson keerthi untuk menentukan titik temu *convex hull* antara dua buah *convex shapes?*

3.2. Analisis Sistem

Aplikasi yang dibangun ini adalah simulasi pembelajaran berbasis *desktop* untuk simulasi pembelajaran fisika momentum dan impulse.

Dalam simulasi ini,bola akan dilemparkan oleh catapult yang mana bola tersebut akan mengarah pada papan yang akan menjadi batasannya, untuk mendeteksi jarak antara bola dan papan serta adanya tumbukan diantara keduanya maka algoritma GJK akan mengatasi hal tesebut, berikut adalah flow chart algoritma GJK yang dapat dilihat pada gambar.



Gambar 3.1 *Flow Chart* Algoritma GJK

Simulasi yang akan dibangun ini merupakan salah satu dari aplikasi yang menggambarkan fisika momentum dan impuls yang terjadi pada catapult yang pada zaman dahulu sering digunakan sebagai alat perang.

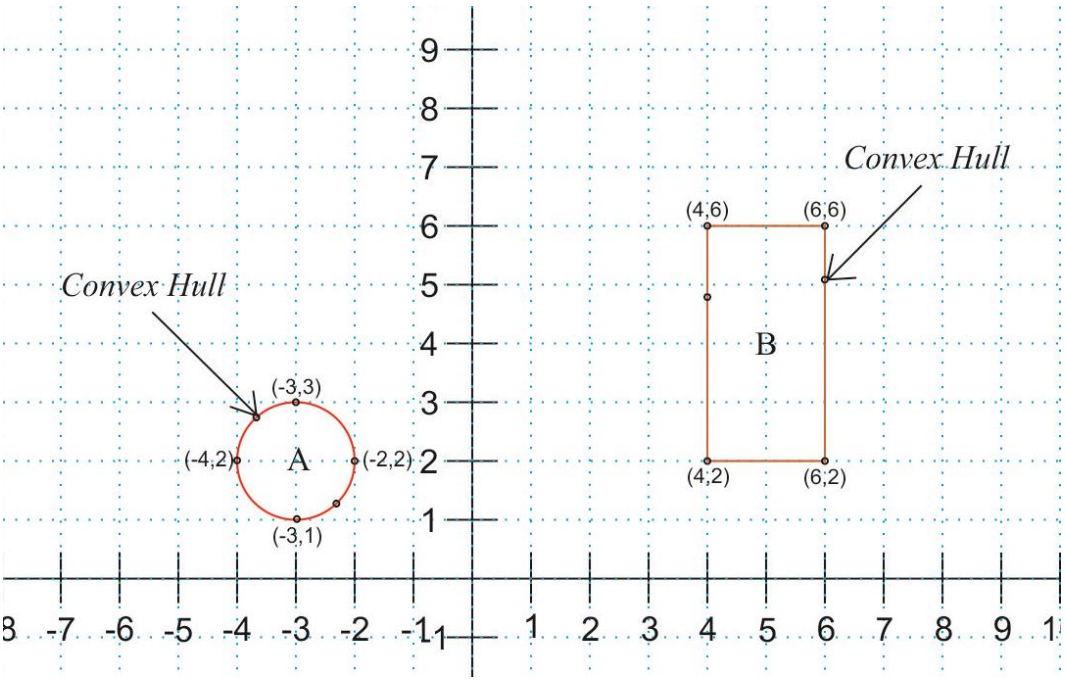
3.3 Analisis Masukan

Pada bagian ini menjelaskan dua jenis data masukan yang diperlukan aplikasi untuk memproses algoritma gilbert jhonson keerthi yaitu *Convex Shapes.*

3.4 Analisis Algoritma Gilbert Jhonson Keerthi

Maksud sebenarnya dari algoritma GJK adalah untuk menentukan jarak terdekat antara dua buah *convex shapes*, namun GJK juga dapat digunakan untuk mengembalikan informasi tumbukan antara dua buah *convex shapes*.[6] Tidak seperti banyak algoritma jarak lainnya, GJK tidak memerlukan data geometri yang disimpan dalam format tertentu, melainkan hanya mengandalkan fungsi dukungan *(Support function)* secara iterasi untuk menghasilkan simplisit terdekat dengan jawaban yang benar menggunakan *minkowski different* dari dua *convex shapes*. Didalam hal ini algoritma GJK digunakan untuk mengembalikan informasi tumbukan antara dua buah objek benda 2D yang kemudian akan berlanjut pada berjalannya hukum momentum dan impuls. Berikut adalah penjelasan dari algoritma GKJ.

Misalkan dalam simulasi momentum dan impulse terdapat 2 buah *convex shapes* yaitu A sebagai peluru dan B sebagai tembok atau penghalang.



Gambar 3.2 *Convex Shape* A dan *Convex Shape* B

GJK menggunakan konstruksi matematika tertentu yang disebut *minkowski different*. *Minkowski different* dari dua buah *convex shape* membentuk *convex shape baru*. *Convex Shape* baru didefinisikan sebagai menambahkan setiap *Convex Hull* A ke setiap *convex hull* di B.

Hal pertama yang dilakukan adalah mendeteksi apakah kedua objek dalam kondisi bertumbukan atau tidak,dengan menggunakan kosep *minkowski different* yaitu diambil titik terluar sebagai perhitungan. Konsep *minkowski different* dapat dihitung dengan menggunakan perhitungan sebagai berikut:

$C = A - B = \{a - b | a \in A, b \in B\}$

Ket : C = *Convex Shape* C

A = *Convex Shape* Peluru

B = *Convex Shape* Tembok

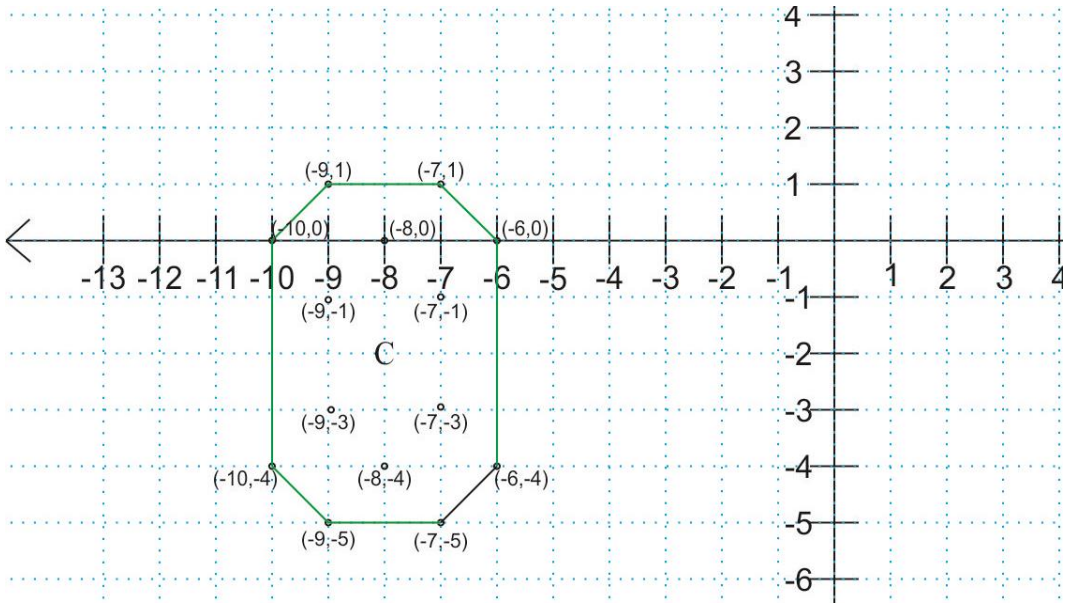
a = *Convex Hull* A

b = *Convex Hull* B

Tabel 3.1 Hasil perhitungan *minkowski different*

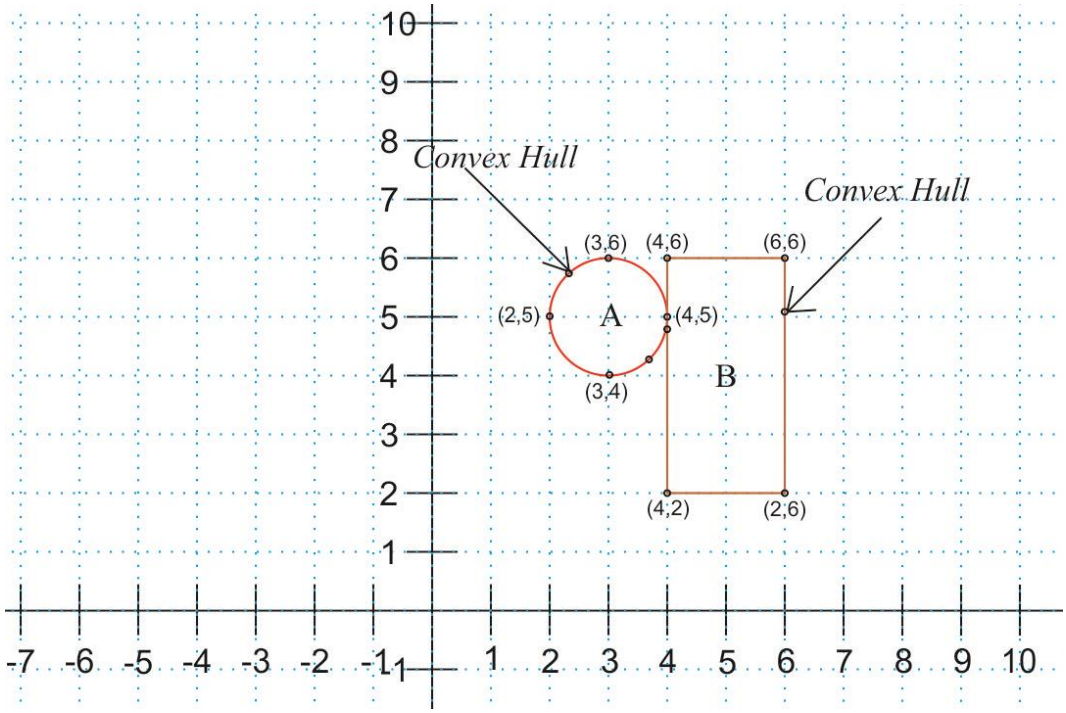
Convex Hull A	Convex Hull B	Convex Hull C (C= A – B)
(-3,1)	(4,2)	(-7,-1)
	(6,2)	(-9,-1)
	(6,6)	(-9,-5)
	(4,6)	(-7,-5)
(-2,2)	(4,2)	(-6,0)
	(6,2)	(-8,0)
	(6,6)	(-8,-4)
	(4,6)	(-6,-4)
(-3,3)	(4,2)	(-7,1)
	(6,2)	(-9,1)
	(6,6)	(-9,-3)
	(4,6)	(-7,-3)
(-4,2)	(4,2)	(-8,0)
	(6,2)	(-10,0)
	(6,6)	(-10,-4)
	4,6)	(-8,-4)

Dengan memasukan titik–titik tersebut kedalam koordinat dan menyambungkan titik-titik terluar satu sama yang lainnya sehingga membentuk suatu *convex shapes* yang baru.



Gambar 3.3 *Convex Shape* C berada diluar *origin*

Terlihat bahwa posisi *convex shapes* yang diciptakan dengan menggunakan konsep *minkowski different* tersebut berada diluar *origin* (0,0) sehingga dapat dikatakan bahwa kedua *convex shapes* A dan B tidak dalam kondisi bertumbukan, sekarang akan diambil contoh kedua ketika posisi peluru berubah mendekati tembok lalu dilakukan kembali konsep *minkowski different.*

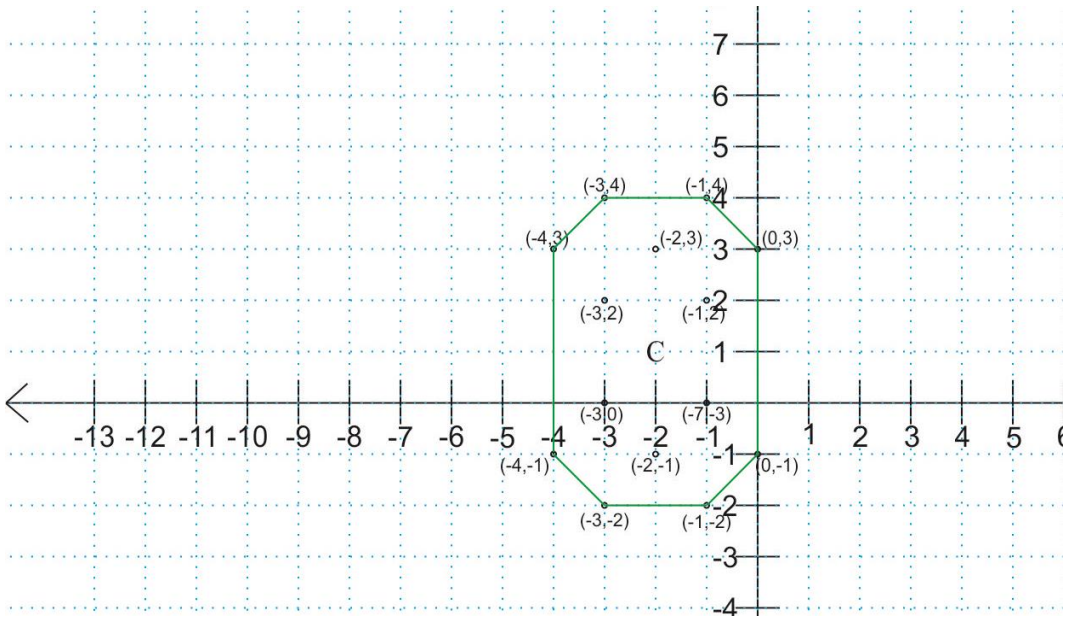


Gambar 3.4 *Convex Shape* A dan B bertumbukan

Berikut adalah perhitungan *minkowski different* pada saat *convex shape* A dan B bertumbukan.

Tabel 3.2 Hasil perhitungan *minkowski different*

Convex Hull A	Convex Hull B	Convex Hull C (C= A – B)
(3,4)	(4,2)	(-1,2)
	(6,2)	(-3,2)
	(6,6)	(-3,-2)
	(4,6)	(-1,-2)
(2,5)	(4,2)	(-2,3)
	(6,2)	(-4,3)
	(6,6)	(-4,-1)
	(4,6)	(-2,-1)
(3,6)	(4,2)	(-1,4)
	(6,2)	(-3,4)
	(6,6)	(-3,0)
	(4,6)	(-1,0)
(4,5)	(4,2)	(0,3)
	(6,2)	(-2,3)
	(6,6)	(-2,-1)
	4,6)	(0,-1)



Gambar 3. 5 *Convex Shape* C berada didalam *origin*

Terlihat bahwa posisi *convex shapes* yang diciptakan dengan menggunakan konsep *minkowski different* tersebut berada didalam *origin* (0,0) atau bersentuhan dengan *origin* sehingga dapat dikatakan bahwa kedua *convex shapes* A dan B berada dalam kondisi bertumbukan.

Dengan ini telah mendapatkan titik yang di tepi bentuk *Minkowski Different* segala arah. Selanjutnyamenggunakan algoritma GJK dengan menggunakan *support function* secara iteratif, adapun urutan konsep algoritma GJK dapat dihitung dengan menggunakan perhitungan sebagai berikut:

d = …

p1 = support(A,B, d)

p2 = support(A,B, -d)

AB = p2 – p1

AO = ORIGIN – p1

d = (AB x AO) x AB

(AB x AO) x AB = AO(AB.dot(AB)) – AB(AB.dot(AO))

p3 = support(A,B, d)

Keterangan :

d = arah vektor *minkowski different*

p1 = titik point pertama

p2 = titik point kedua

p3 = titik point ketiga

A = titik point *convex shape* A

B = titik point *convex shape* B

ORIGIN = titik point (0 , 0)

Perhitungan *simplex* dilakukan secara iteratif sehingga *simplex* semakin dekat dengan mengandung origin. Kemudian perlu memeriksa kondisi iterasi apakah simplex mengandung origin? Berikut perhitungan algoritma GJK dengan menggunakan kerangka pada gambar 3.3 dengan arah vektor awal (1 , 0).

d = (1,0)

p1 = support(A , B , d)

= (4 , 5) – (4 , 6)

= (0 , -1)

Terlihat dengan inisialisasi arah vector awal (1,0) didapatkan titik point awal (0,-1) . Kemudian dimulai pengulangan :

Iterasi pertama

p2 = support(A , B , -d)

= (2 , 5) – (6, 2)

= (-4 , 3)

AB = (-4 , 3) – (0 , -1) = (-4 , 4)

AO = (0 , 0) – (-4 , 3) = (4 , -3)

AB.dot(AB) = (xAB . xAB) + (yAB . yAB)

= (-4 . -4) + (4 . 4)

= 16 + 16 = 32

AB.dot(AO) = (xAB . xAO) + (yAB . yAO)

= (-4 . 4) + (4 . -3)

= -16 + (-12) = -28

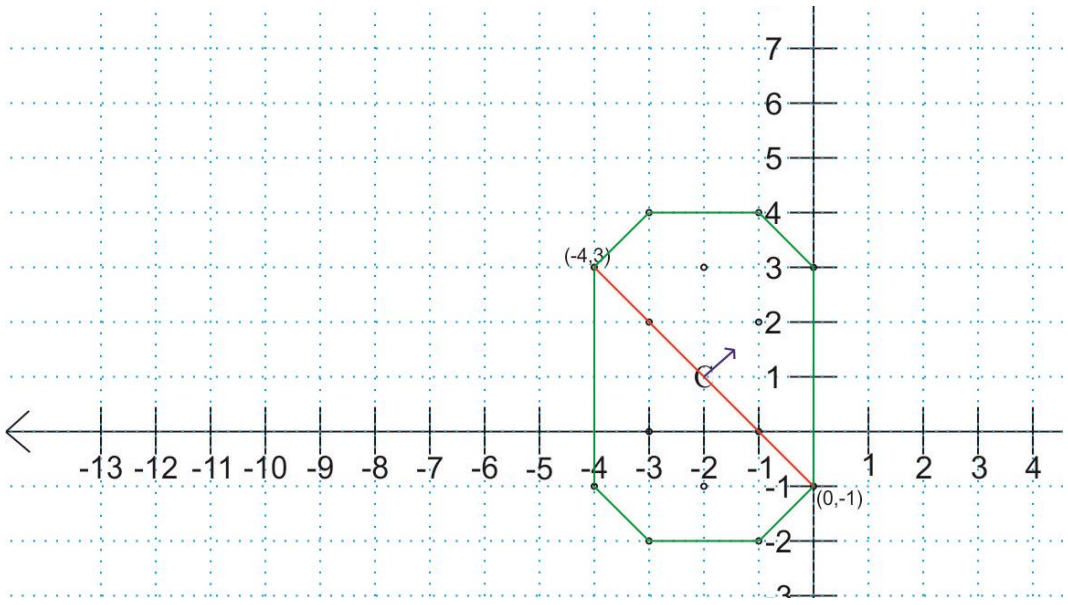
(AB x AO) x AB = AO(32) – AB(-28)

= (128 – 96) – (112 , -112)

= (16 , 16)

= (1 , 1)

d = (1 , 1)



Gambar 3.6 Iterasi pertama

Gambar 3.6 menunjukkan simpleks yang dihasilkan setelah iterasi pertama. Simpleks ditunjukan dengan garis berwarna merah, simpleks dengan arah berikutnya berwarna biru menunjuk tegak lurus terhadap garis ke arah *origin*.

Iterasi kedua

p3 = support(A , B , d)

= (3 , 6) – (4 , 2) = (-1 , 4)

AB = (-1 , 4) – (0 , -1) = (-1 , 5)

AO = (0, 0) – (-1 , 4) = (1 , -4)

AB.dot(AB) = (xAB . xAB) + (yAB . yAB)

= (-1 . -1) + (5 . 5)

= 1 + 25 = 26

AB.dot(AO) = (xAB . xAO) + (yAB . yAO)

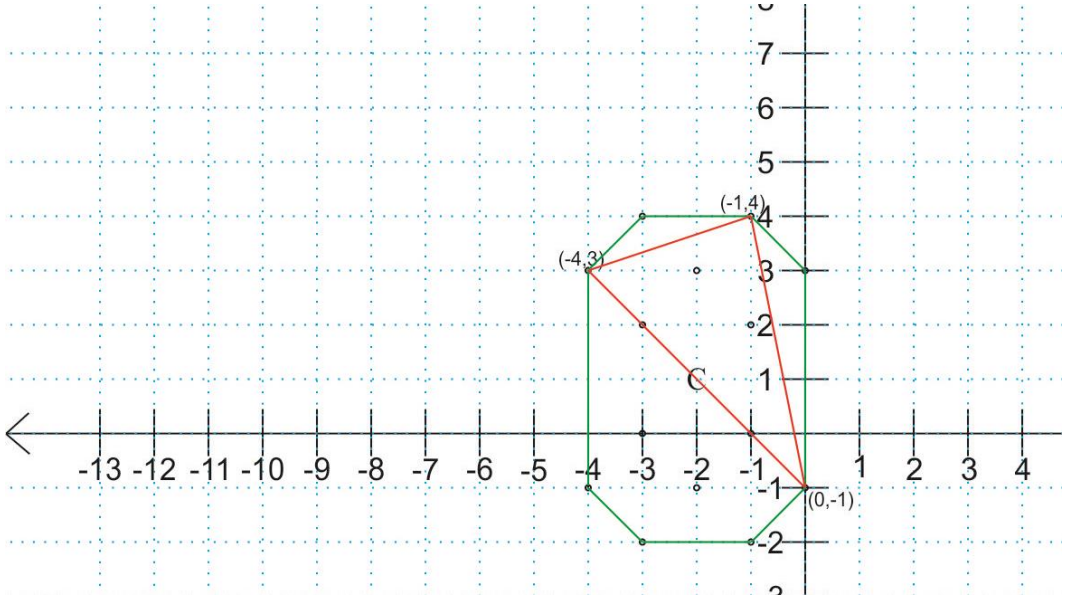
= (-1. 1) + (5 . -4)

= -1 + (-20) = -21

(AB x AO) x AB = AO(26) – AB(-21)

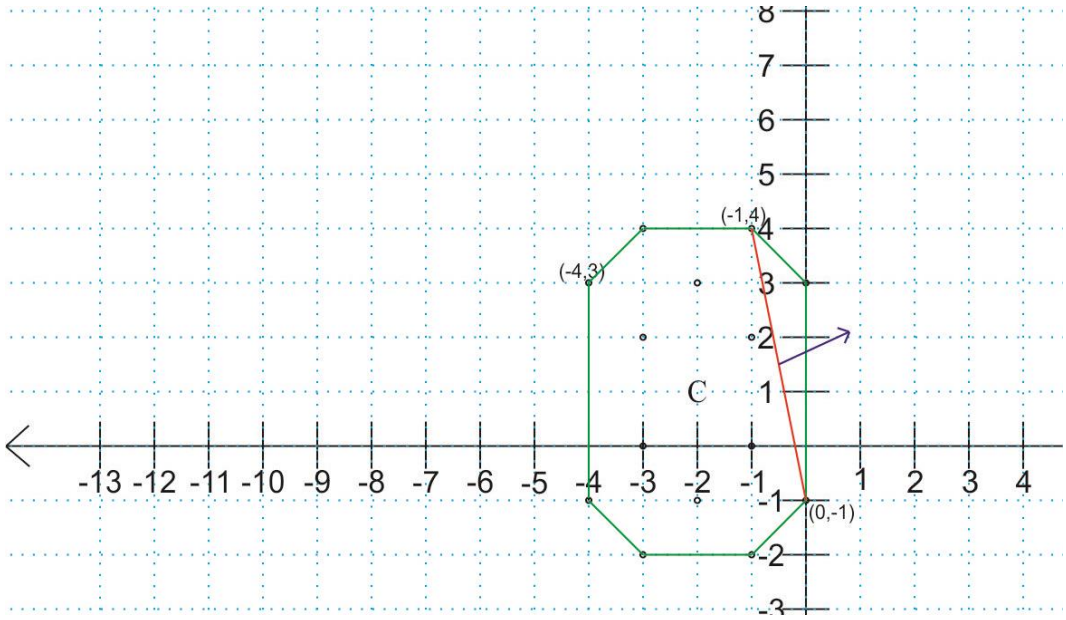
= (26 , -104) - (21, -105)

= (5 , 1) = (1 , 0.2)



Gambar 3.7a Iterasi kedua

Gambar 3.7a menunjukkan simpleks yang dihasilkan setelah iterasi kedua belum melewati origin namun masih belum bisa menyimpulkan bahwa *convex shapes* A dan *convex shapes* B tidak bertumbukan atau berpotongan.



Gambar 3.7b Iterasi kedua dan arah vektor

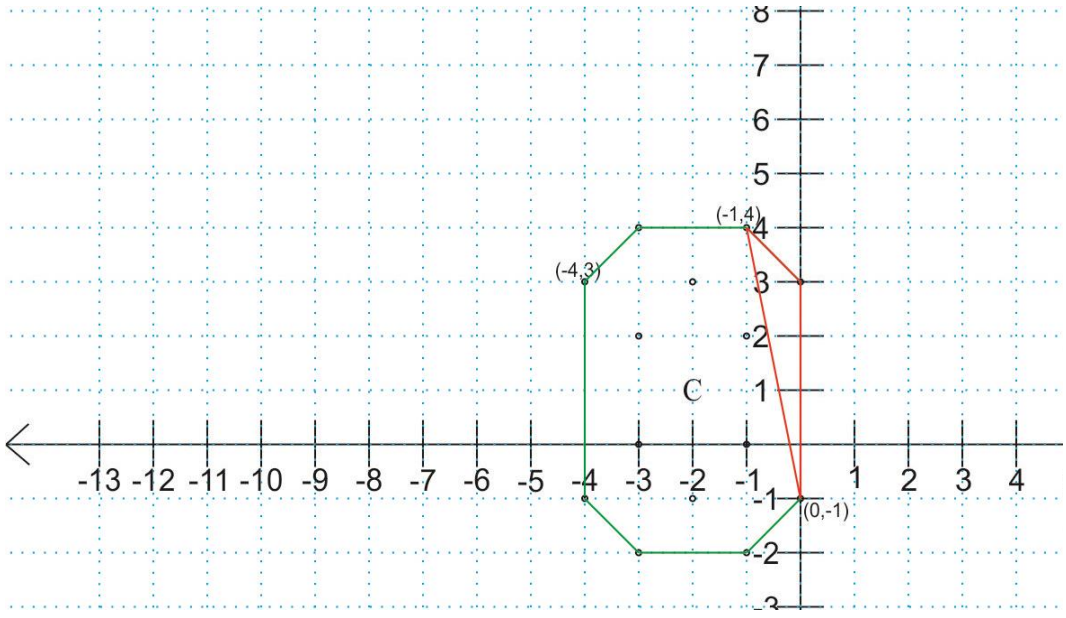
Pada iterasi kedua dihapus titik (-4, 3) karena semua yang dibutuhkan adalah 3 poin setiap saat menambahkan titik baru pada awal setiap iterasi.

Iterasi ketiga

d = (1 , 0.2)

p4 = support(A ,B , d)

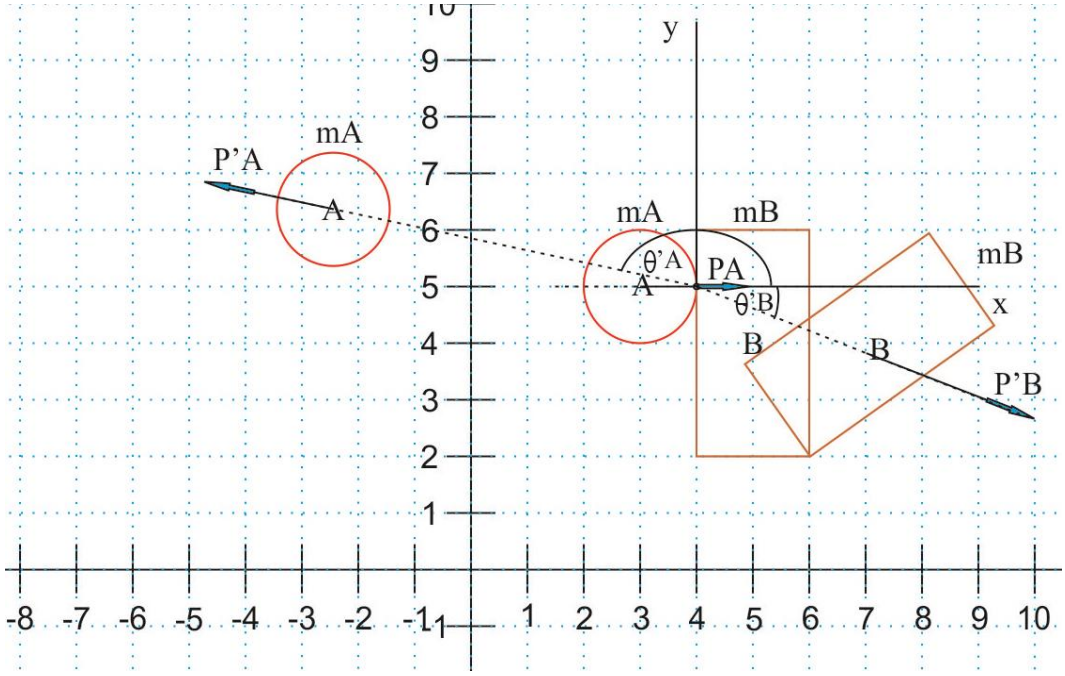
= (4 , 5) – (4 , 2) = (0 , 3)



Gambar 3.8 Iterasi ketiga

Gambar 3.8 menunjukkan simpleks yang dihasilkan setelah iterasi ketiga telah melewati origin sehingga dapat disimpulkan bahwa bahwa *convex shapes* A dan *convex shapes* B bertumbukan atau berpotongan.

Pada saat diketahui *convex shapes* A dan *convex shapes* B bertumbukan, kedua objek akan mengalami impuls yang mengubah besaran momentum di kedua benda



Gambar 3.9 Momentum dan Impuls

Gambar 3.9 menunjukan *convex shapes* A (bola) bertumbukan dan memberikan sebuah nilai besaran momentum kepada *convex shapes* B (tembok) sehingga kedua benda mengalami besaran impuls yang menghasilkan perubahan nilai besaran momentum dikedua buah benda.

3.5 Analisis Kebutuhan Non Fungsional

Analisis kebutuhan non fungsional merupakan analisis untuk menentukan spesifikasi kebutuhan sistem.Spesifikasi kebutuhan sistem meliputi analisis perangkat keras atau hardware, analisis perangkat lunak atau software, dan analisis pengguna.

3.5.1 Analisis Kebutuhan Perangkat Lunak

Analisis perangkat lunak digunakan untuk mengetahui spesifikasi perangkat lunak yang digunakan untuk menjalankan simulasi fisika momentum dan impuls, spesifikasi perangkat lunak yang digunakan untuk menjalankan simulasi ini adalah sebagai berikut :

1. Sistem Operasi Windows 7 (32 bit dan 64 bit)
1. Adobe Flash CS 4

Sedangkan perangkat lunak yang dibutuhkan untuk memainkan simulasi ini adalah *adobe flash player*, dengan spesifikasi sebagai berikut [14] :

2. Windows 7 (32 bit dan 64 bit)
3. Mozilla Firefox 17 atau yang lebih baru.

3.5.2 Analisis Kebutuhan Perangkat Keras

Analisis perangkat keras digunakan untuk mengetahui spesifikasi perangkat keras yang digunakan untuk menjalankan simulasi fisika momentum dan impuls perangkat keras yang digunakan untuk menjalankan simulasi ini adalah :

1. *Processor* @ 1.83GHz (2 CPUs)
2. *Memory* (RAM) 1536MB
3. *Display Memory* 256 MB
4. *Display Mode*1280 x 800 (32 bit) (60Hz)
5. *Input DeviceMouse* dan *Keyboard*

Sedangkan untuk spesifikasi perangkat keras yang dibutuhkan untuk dapat menjalankan simulasi ini adalah sebagai berikut [14] :

1. Procesor2.33GHz atauprosesor x86-kompatibel lebih cepat.
2. 512MB RAM (1GB RAM direkomendasikan untuk netbook); 128MB memori grafis

3.5.3 Analisis Pengguna

Analisis pengguna digunakan untuk mengetahui spesifikasi pengguna agar dapat menjalankan program ini.Spesifikasi minimal bagi pengguna dapat dilihat pada tabel 3.3.

Tabel 3.3 Spesifikasi Pengguna

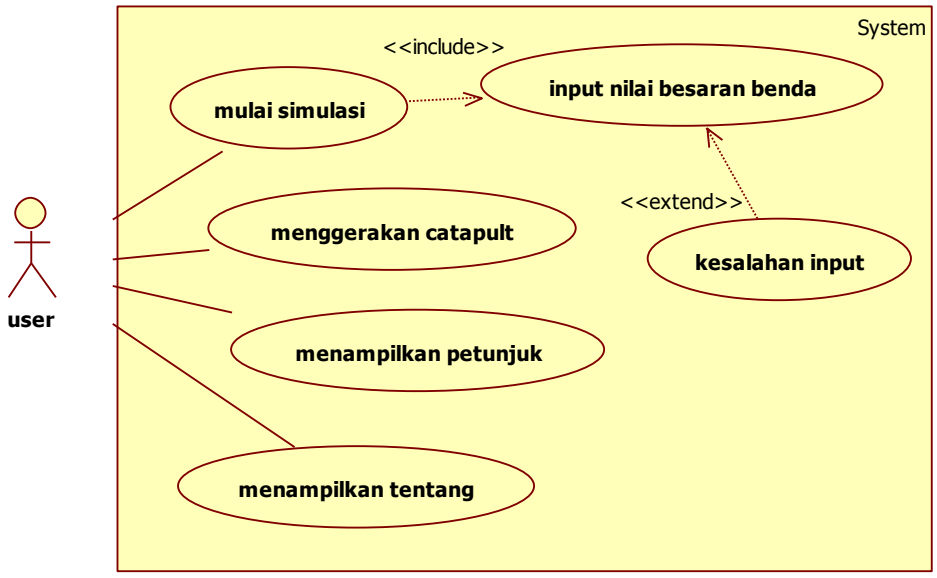
Karakteristik Fisik Pengguna	
Umur	14 tahun keatas
Gender	Laki-laki dan Perempuan
Disabilities	Tidak buta dan tidak mempunyai cacat fisik seperti cacat tangan
Pengetahuan dan Pengalaman Pengguna	
Computer literacy	Dapat menggunakan dan mengoperasikan computer
Education	14 tahun keatas

3.6 Analisis Kebutuhan Fungsional

Analisis kebutuhan fungsional merupakan analisis untuk menggambarkan sistem yang akan dibangun. Analisis kebutuhan fungsional dimodelkan dengan menggunakan UML, tahap-tahap pemodelannya antara lain adalah *use case diagram, use case scenario, activity diagram, class diagram* dan *sequence diagram*.

3.6.1 *Use Case Diagram*

Use case diagram merupakan pemodelan untuk kelakuan (*behavior*) sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat . Berikut adalah *use case diagram* Program Pembelajaran Fisika Momentum dan Impulse dapat dilihat pada gambar 3.10.



Gambar 3.10 *Use Case* Diagram

Use case terdiri dari tiga bagian yaitu definisi *Actor*, definisi *use case*, dan skenario *use case*.

3.6.1.1 Definisi *Actor*

Definisi aktor berfungsi untuk menjelaskan aktor yang terdapat pada *use case diagram*. Deskripsi pendefinisian aktor Sistem Pembelajaran Fisika Momentum dan Impulse dapat dilihat pada tabel 3.4

Tabel 3.4 Definisi aktor pada Program Pembelajaran Fisika Momentum dan Impulse

No	Aktor	Deskripsi
1	User	Orang yang menjalankan simulasi

3.6.1.2 Definisi *Use Case*

Definisi *use case* berfungsi untuk menjelaskan proses yang terdapat pada *use case diagram*. Deskripsi pendefinisian use case pada Program Pembelajaran Fisika Momentum dan Impulse dapat dilihat pada tabel 3.5

Tabel 3.5 Definisi use case pada Program Pembelajaran Fisika Momentum dan Impulse

No	Use Case	Deskripsi
1	Mulai simulasi	Proses untuk memulai simulasi pembelajaran fisika momentum dan impulse
2	Menampilkan petunjuk	Proses untuk menampilkan cara menggunakan simulasi pembelajaran fisika momentum dan impulse
3	Menampilkan tentang	Menampilkan info simulasi dan profil
4	Memasukan nilai besaran benda	*User* menginputkan nilai besaran kepada dua buah objek benda yaitu peluru dan tembok
5	Menjalankan catapult	*User* menggerakan catapult untuk menentukan jarak benda yang akan diluncurkan terhadap dinding

3.6.1.3 Skenario *Use Case*

Use case Scenario mendeskripsikan urutan langkah-langkah dalam proses bisnis baik yang dilakukan aktor terhadap sistem maupun yang dilakukan oleh sistem terhadap aktor. Berdasarkan *use case* diagram, maka *use case scenario* untuk *prototype* Program Pembelajaran Fisika Momentum dan Impulse terdiri dari mulai simulasi, petunjuk , tentang , *input* nilai besaran benda, menggerakan catapult. Berikut adalah *use case scenario* Program Pembelajaran Fisika Momentum dan Impulse :

1. *Use Case Scenario* Mulai simulasi

Use case scenario ini menjelaskan interaksi antara aktor dengan *use case* mulai simulasi yang dijelaskan pada table 3.6

Tabel 3.6 *Use Case Scenario* Mulai Simulasi

Identifikasi		
Nama	Mulai simulasi	
Tujuan	Memulai simulasi	
Deskripsi	Proses untuk memulai simulasi	
Aktor	*User*	
Skenario Utama		
Kondisi awal	*User* berada di layar simulasi	
Aksi Aktor		**Reaksi Sistem**
Skenario Normal		
a. Memilih menu simulasi		
		b. Menampilkan jendela simulasi
c. Mengklik tombol mulai		
		d. Mengeksekusi simulasi, catapult dijalankan dan melontar peluru sesuai dengan masukan nilai parameter
		e. Peluru yang dilontarkan akan menghitung kondisi secara iterasi apakah peluru dalam keadaan bertumbukan dengan tembok atau tidak dengan aturan GJK dan pendekatan *minkowski different* dengan cara menghitung posisi *simplex* berada pada origin atau tidak
		f. Jika kondisi terpenuhi *simplex* berada dalam *origin* maka diketahui peluru telah bertumbukan dengan tembok

	g. Menghitung perubahan momentum pada peluru dan tembok sesuai dengan arah benda dan parameter masukan
	h. Menampilkan kondisi peluru dan tembok saat terjadi momentum dan impuls
	i. Jika kondisi tidak terpenuhi simplex tidak berada dalam origin maka peluru tidak bertumbukan dengan tembok dan mengulang iterasi
Kondisi Akhir	Terjadi perubahan besaran momentum diantara peluru dan tembok

2. *Use Case Scenario* Input Nilai Besaran Benda

Use case scenario ini menjelaskan interaksi antara aktor pengguna dengan *use case* menginputkan nilai besaran benda yang dijelaskan pada table 3.7

Tabel 3.7 *Use Case Scenario* input nilai besaran benda

Identifikasi	
Nama	Input nilai besaran benda
Tujuan	Input nilai besaran benda
Deskripsi	Proses untuk menginputkan nilai besaran benda
Aktor	*User*
Skenario Utama	
Kondisi awal	*User* berada di layar simulasi
Aksi Aktor	**Reaksi Sistem**
Skenario Normal	
a. Input parameter nilai besaran benda peluru dan tembok	
b. Klik tombol input	

	c. Memproses nilai besaran inputan
	d. Jika input dengan nilai masukan benar maka Memproses dan menampilkan peluru dan tembok sesuai dengan nilai besaran yang telah diinputkan
	e. Jika klik tombol input dengan nilai masukan salah satu nilai besaran benda tidak sesuai atau tidak benar
	f. memproses dan menampilkan pesan kesalahan
Kondisi Akhir	Menampilkan peluru dan tembok jika benar

3. *Use Case Scenario* Menggerakan *Catapult*

Use case scenario ini menjelaskan interaksi antara aktor pengguna dengan *use case* menggerakan *catapult* yang dijelaskan pada tabel 3.8

Tabel 3.8 *Use Case Scenario* Menggerakan *catapult*

Identifikasi	
Nama	Menggerakan *catapult*
Tujuan	Menggerakan *catapult*
Deskripsi	Proses untuk menggerakan *catapult*
Aktor	*User*
Skenario Utama	
Kondisi awal	*User* telah mengimputkan nilai besaran benda
Aksi Aktor	**Reaksi Sistem**
Skenario Normal	
a. Mengatur posisi *catapult* dengan menekan tombol	

arah panah kanan atau kiri dari *keyboard*	
	b. Jarak antara catapult dengan dengan tembok akan akan lebih jauh atau lebih dekat
Kondisi Akhir	Posisi catapult berubah

4. *Use Case Scenario* Tentang

Use case scenario ini menjelaskan interaksi antara aktor pengguna dengan *use case* tentang yang dijelaskan pada Tabel 3.9

Tabel 3.9 *Use Case Scenario* Tentang

Identifikasi	
Nama	Tentang
Tujuan	Menampilkan info aplikasi
Deskripsi	Proses info aplikasi
Aktor	Pengguna
Skenario Utama	
Kondisi awal	Pengguna berada di layar utama
Aksi Aktor	**Reaksi Sistem**
Skenario Normal	
a. Memilih menu tentang	
	b. Menampilkan info aplikasi
Kondisi Akhir	Menampilkan informasi aplikasi

5. *Use Case Scenario* Petunjuk

Use case scenario ini menjelaskan interaksi antara aktor pengguna dengan *use case petunjuk* yang dijelaskan pada tabel 3.10

Tabel 3.10 *Use Case Scenario* Petunjuk

Identifikasi	
Nama	Petunjuk
Tujuan	Menampilkan info petunjuk

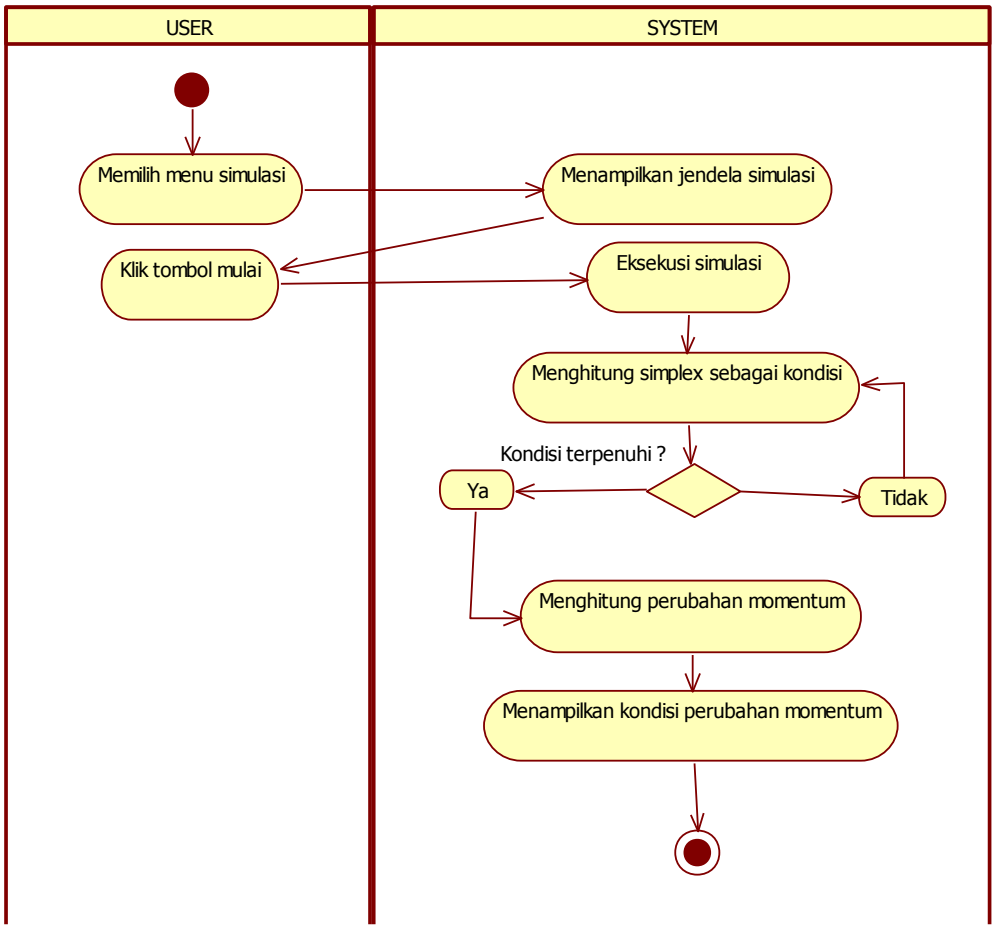
Deskripsi	Proses info petunjuk	
Aktor	Pengguna	
Skenario Utama		
Kondisi awal	Pengguna berada di layar utama	
Aksi Aktor		**Reaksi Sistem**
Skenario Normal		
a. Memilih menu petunjuk		
		b. Menampilkan info petunjuk
Kondisi Akhir	Menampilkan informasi petunjuk	

3.6.2. *Activity Diagram*

Diagram aktivitas atau *activity diagram* menggambarkan *work flow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak. Berikut adalah *activity diagram* dari *prototype* Program Pembelajaran Fisika Momentum dan Impulse :

1. *Activity Diagram* Mulai simulasi

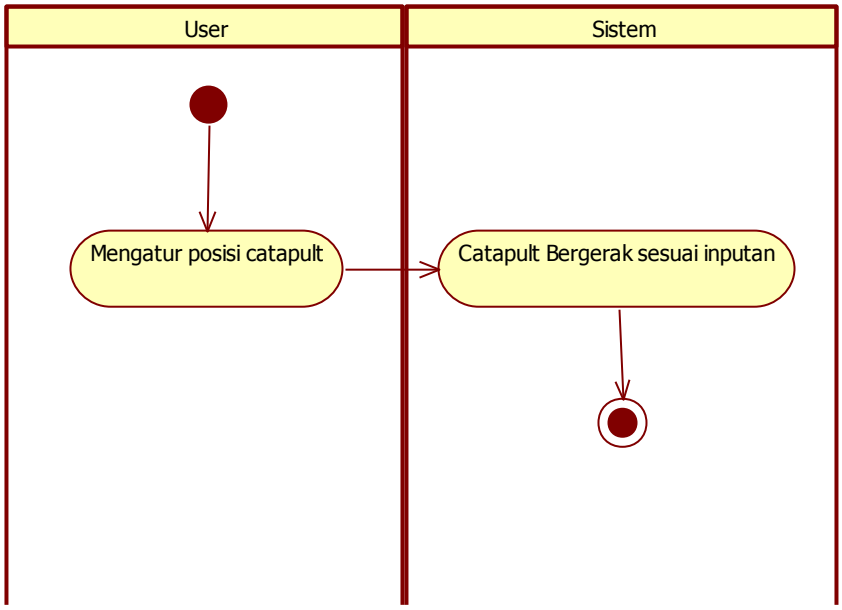
Pada proses ini sistem akan memulai menampilkan *interface* simulasi Pembelajaran Fisika Momentum dan Impulse . *Activity diagram* mulai dapat dilihat pada gambar 3.10



Gambar 3.11 *Activity Diagram* mulai simulasi

2. *Activity Diagram* menggerakan catapult

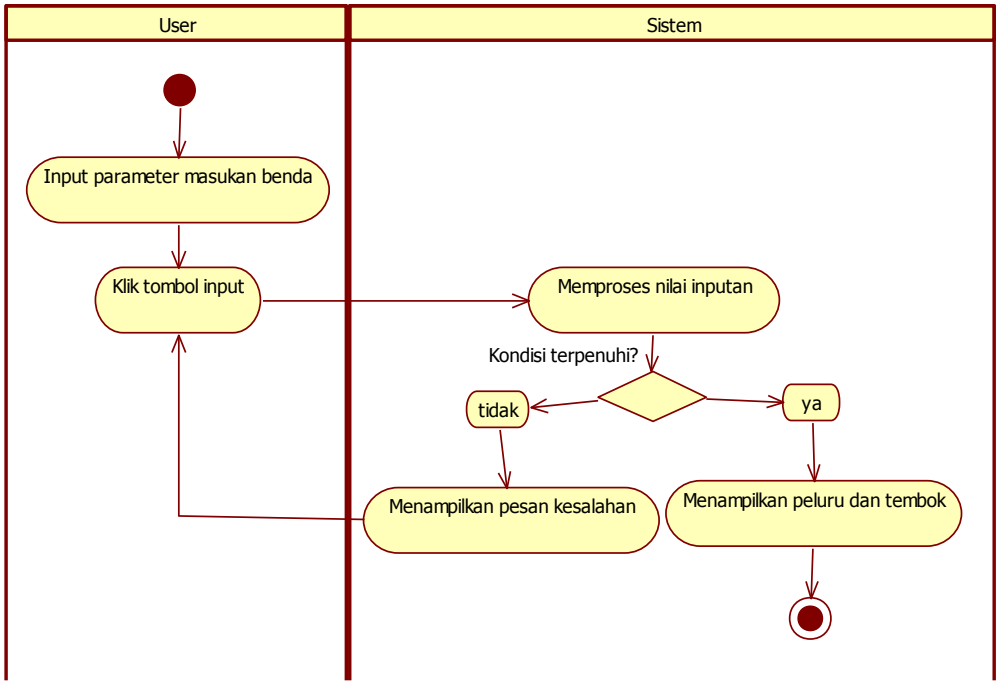
Pada proses ini *user* akan menggerakan catapult. *Activity diagram* menggerakan catapult dapat dilihat pada gambar 3.11



Gambar 3.12 *Activity Diagram* menggerakan catapult

3. *Activity Diagram* Input Nilai Besaran Benda

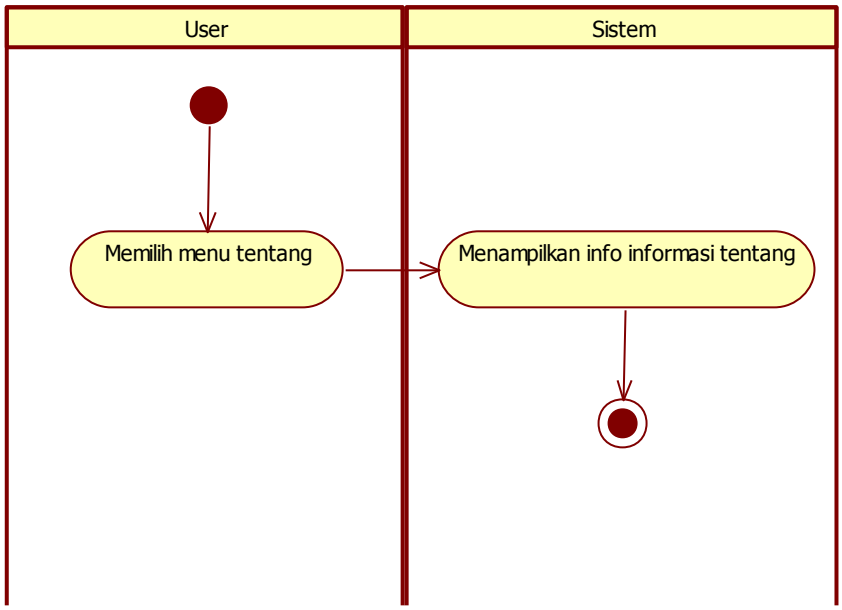
Pada proses ini *user* akan menentukan massa peluru. *Activity diagram* input nilai besaran benda dapat dilihat pada gambar 3.12



Gambar 3.13 *Activity Diagram Input* nilai besaran benda

4. *Activity Diagram* tentang

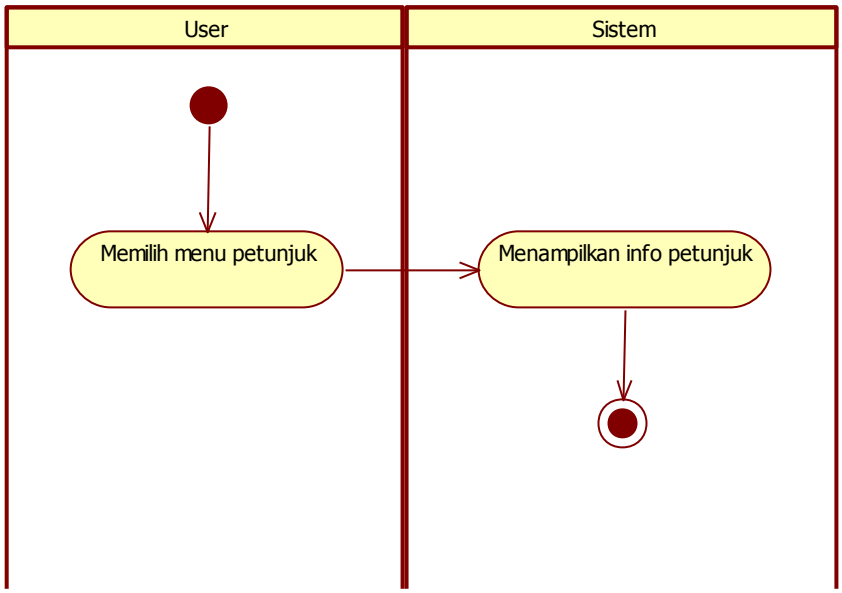
Pada proses ini sistem akan menampilkan *interface* tentang. *Activity diagram* tentang dapat dilihat pada gambar 3.14



Gambar 3.14 *Activity Diagram* tentang

5. *Activity Diagram* Petunjuk

Pada proses ini sistem akan menampilkan *interface* petunjuk. *Activity diagram* petunjuk dapat dilihat pada gambar 3.15

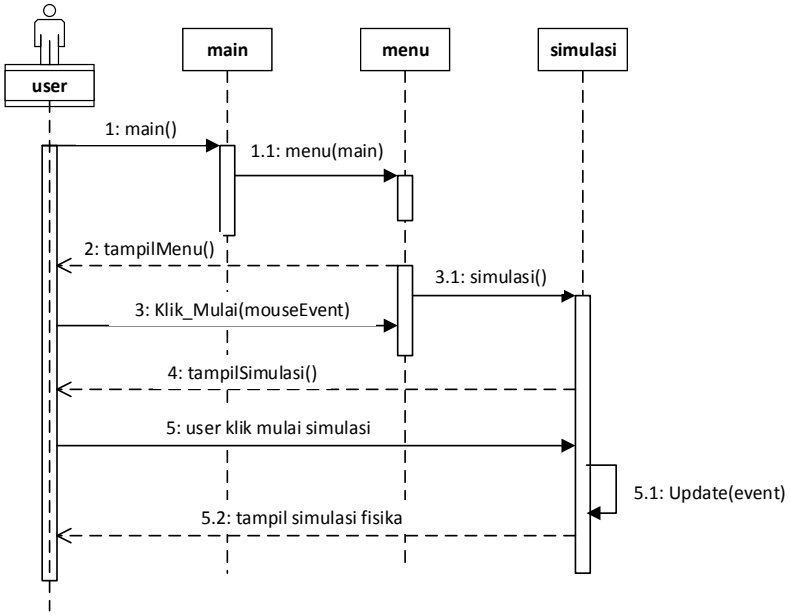


Gambar 3.15 *Activity Diagram* Petunjuk

3.6.*3. Squence Diagram*

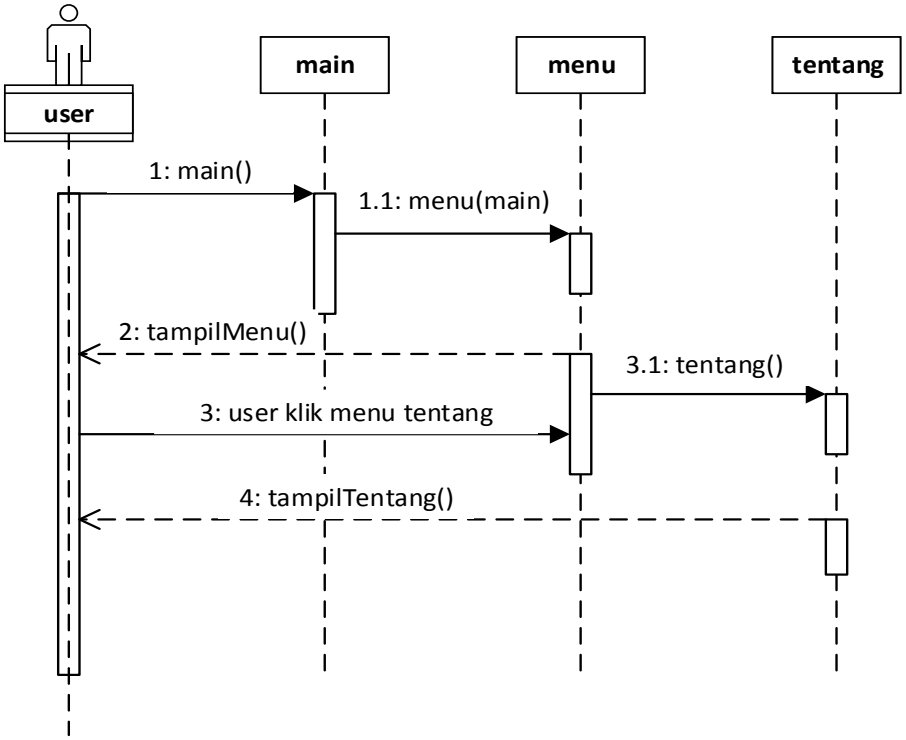
Diagram sekuen menggambarkan kelakuan objek pada use case dengan mendeskripsikan waktu hidup objek dan message yang dikirimkan dan diterima antar objek. Berikut adalah *sequence diagram* dari simulasi fisika momentum dan impuls :

1. *Sequence Diagram* Mulai simulasi



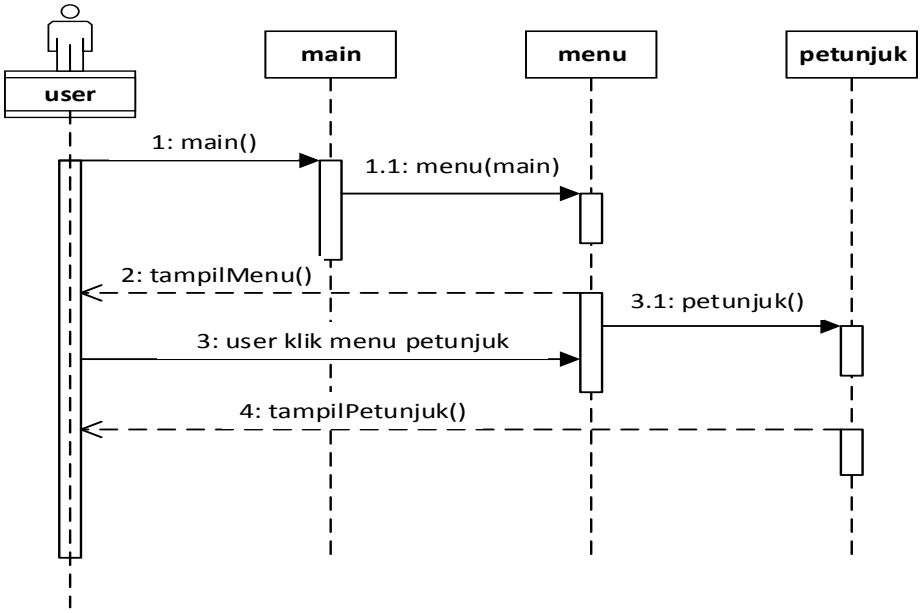
Gambar 3.16 *Sequence Diagram* Mulai Simulai

2. *Sequence Diagram* Tentang



Gambar 3.17 *Sequence Diagram* Tentang

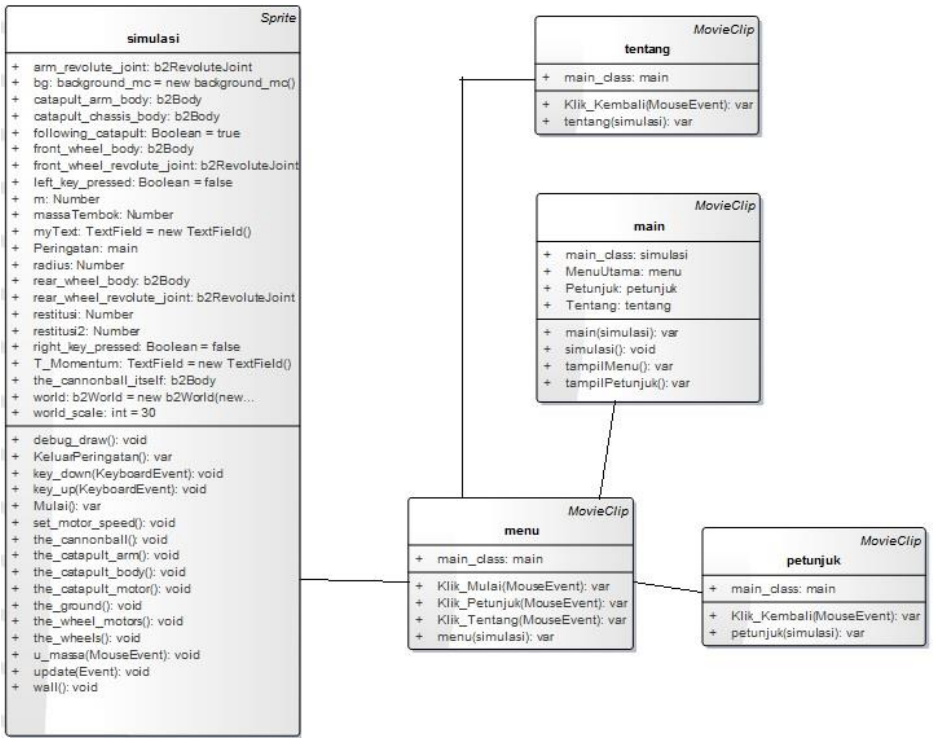
3. *Sequence Diagram* Petunjuk



Gambar 3.18 *Sequence Diagram* Petunjuk

3.6.*4. 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. Berikut adalah *class diagram* dari simulasi fisika momentum dan impuls :



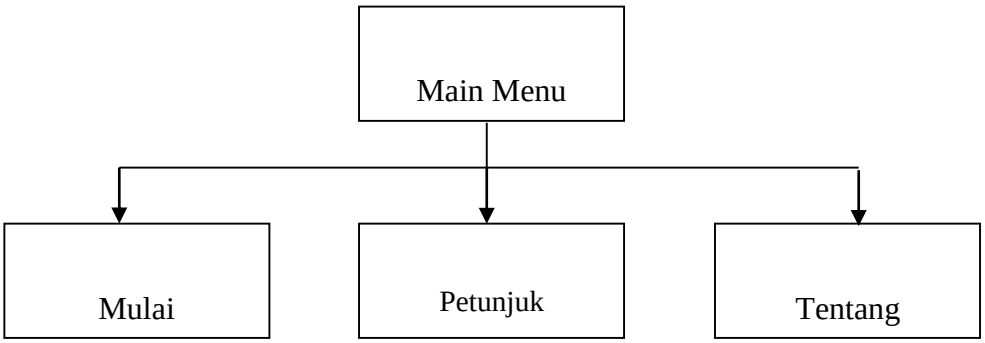
Gambar 3.19 *Class Diagram* simulasi fisika momentum & impuls

3.7 Perancangan Sistem

Perancangan sistem merupakan kegiatan untuk merancang aplikasi yang akan di bangun, tahapannya dimulai dari perancangan struktur menu, perancangan antarmuka, perancangan pesan dan jaringan semantik dari program simulasi fisika momentum dan impulse.

3.7.1 Perancangan Stuktur Menu

Perancangan struktur menu pada program simulasi fisika momentum dan impulse adalah sebagai berikut:

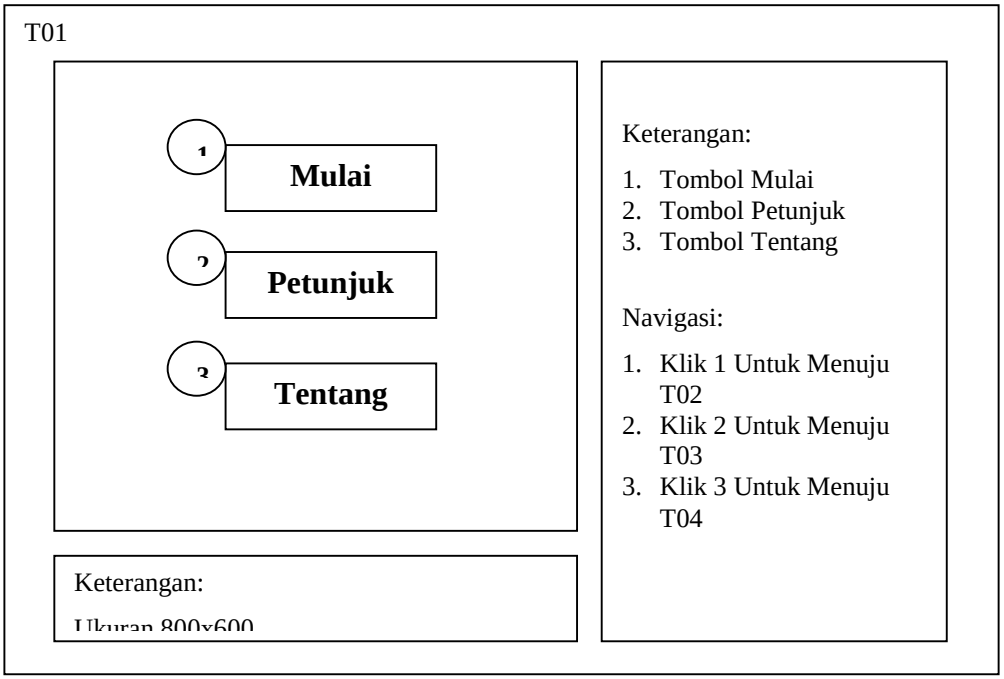


Gambar 3.20 Struktur Menu

3.7.2 Perancangan Antar Muka

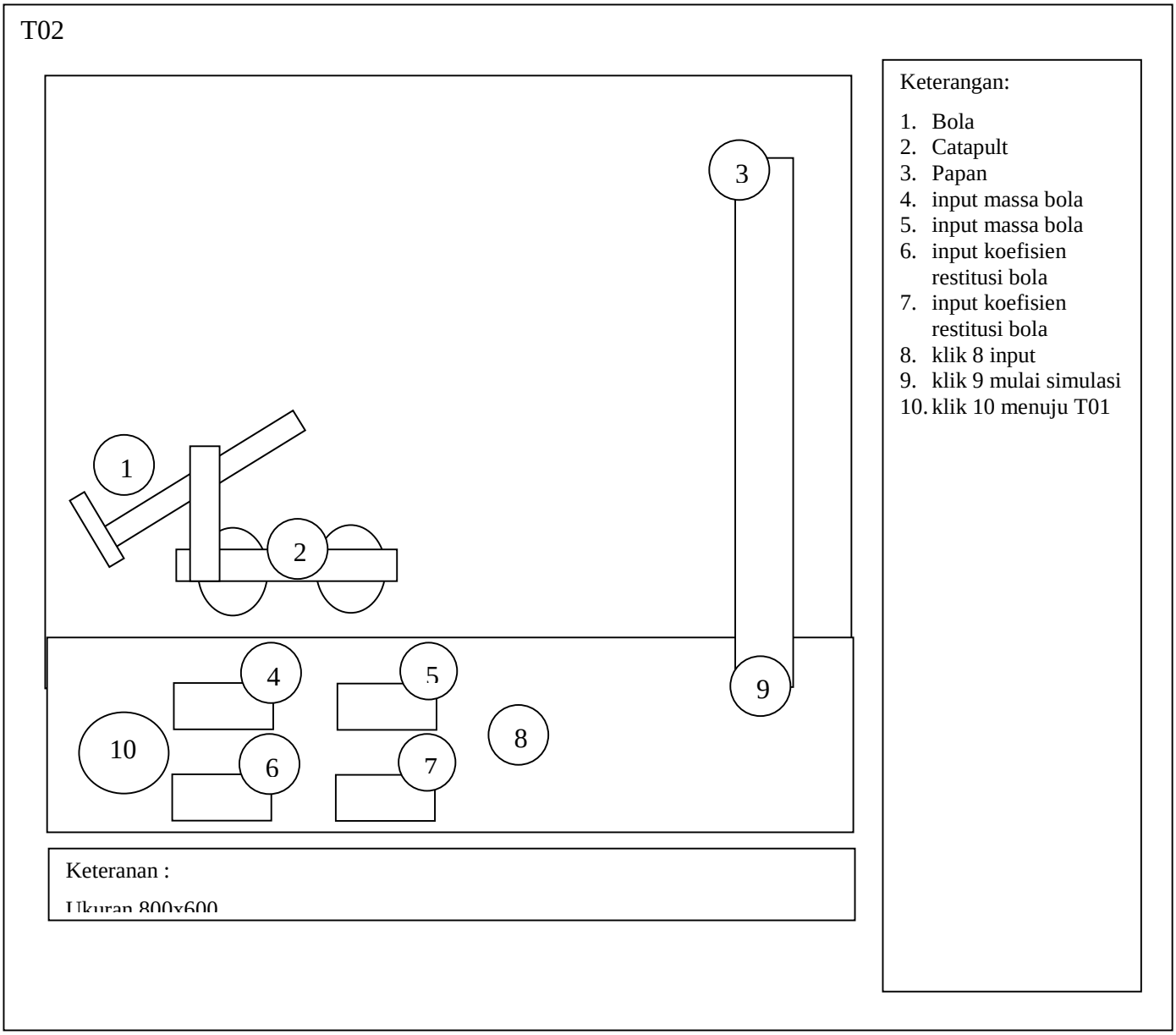
Perancangan antar muka bertujuan untuk memberikan gambaran tentang aplikasi yang akan dibangun, sehingga akan mempermudah dalam mengimplementasikan aplikasi serta akan memudahkan pembuatan aplikasi.

1. Perancangan Antar muka Main Menu



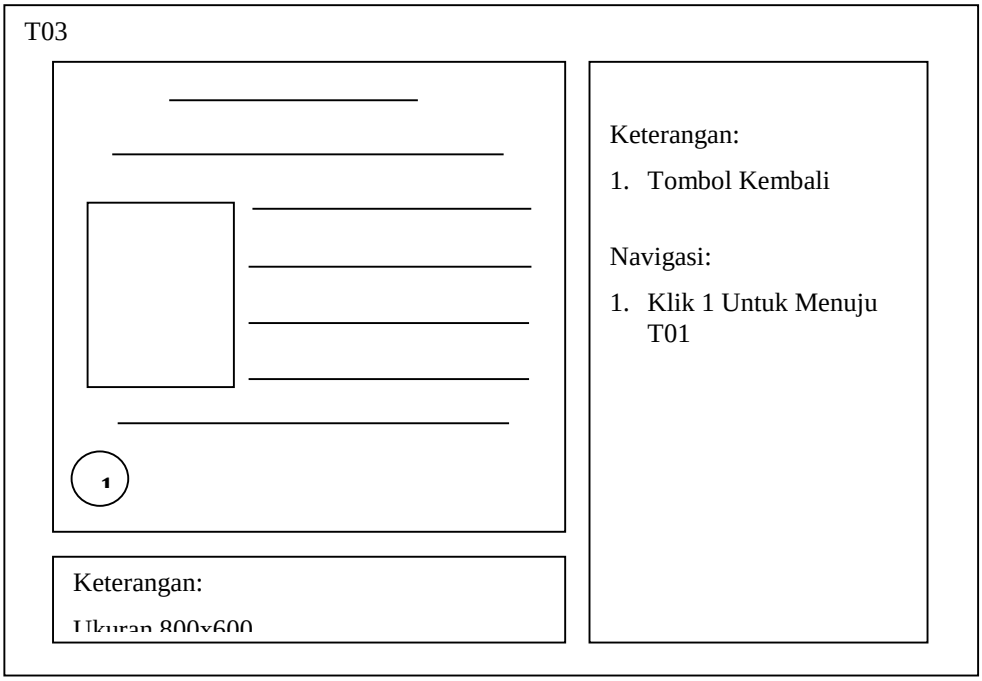
Gambar 3.21 Antar Muka Main Menu

2. Perancangan Antarmuka Mulai



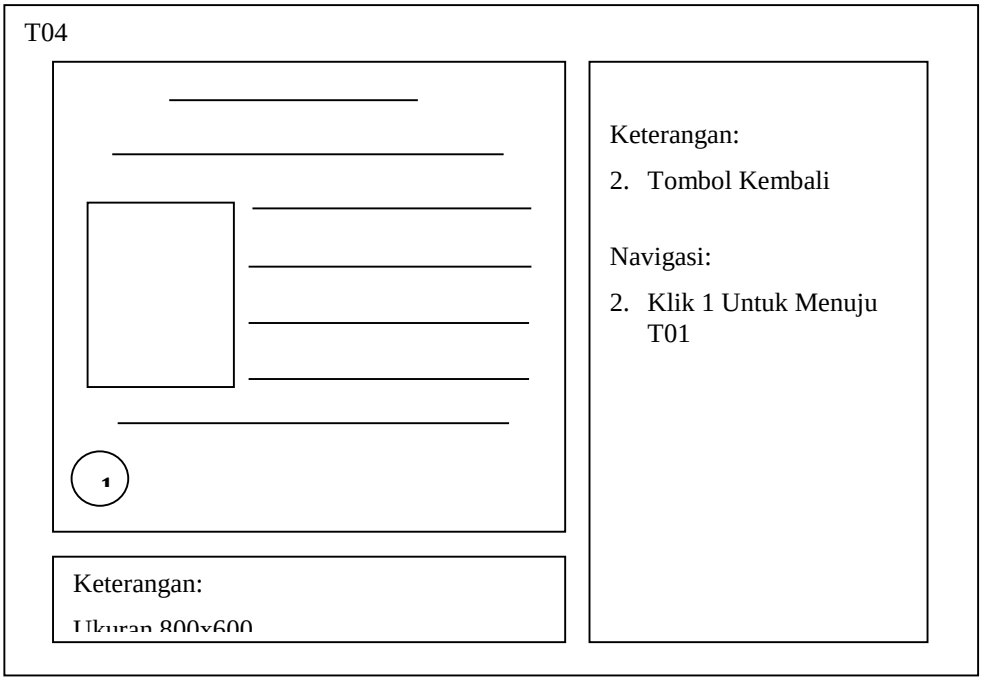
Gambar 3.22 Antar Muka Mulai

3. Perancangan Antarmuka Petunjuk



Gambar 3.23 Antar Muka Petunjuk

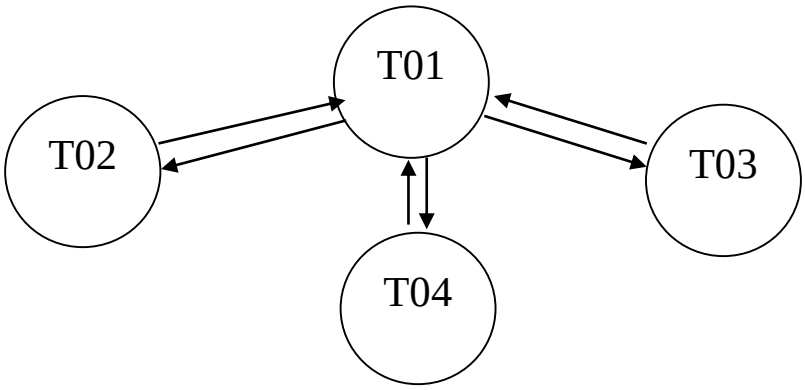
4. Perancangan Antarmuka Tentang



Gambar 3.24 Antar Muka Tentang

3.7.3 Jaringan Semantik

Untuk mengetahui keterhubungan dari setiap tampilan yang dibuat dapat menggunakan jaringan semantik. Berikut adalah jaringan semantik pada simulasi pembelajaran fisika momentum dan impuls.



Gambar 3.25 Jaringan Semantik