Abstract

This document covers the development of new means for modelling and calculating risk areas by Monte Carlo-simulating projectile trajectories in MATLAB.

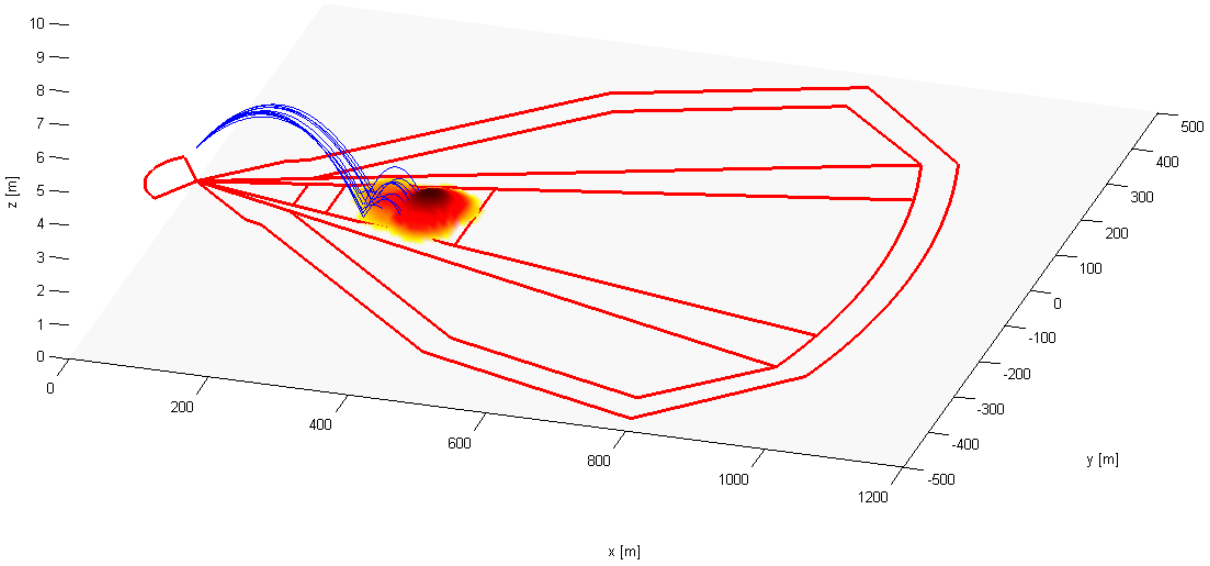
The Swedish Armed Forces safety instruction (*SäkI*) defines a risk area as an area where damages can occur during certain activities. The activity studied here was the firing of projectiles over an arbitrary area. The risk areas defined by *SäkI* give an outlining of the worst case scenario outside of which the risk of damage is negligible.

The new risk areas created are built up of probability values that arise during the Monte Carlo-simulations. As a comparison the risk areas defined in *SäkI* are calculated and displayed together with the simulated risk areas. For this models and methods for calculating projectile trajectories, ricochets and detonation probabilities were defined and used in the Monte Carlo-simulations.

The simulated risk areas confirm that SäkI´s risk areas rather well cover the worst case scenarios and that during regular circumstances a much smaller risk area could be applied.

With the developed software it is possible to vary different parameters to distinguish what characteristics result in the greatest risk areas. By doing so restrictions can be determined for certain projectiles/firing system to reduce risk areas or in earlier stages of development system characteristics can be altered to reduce the overall risk area.

Examensarbete i
Modellering och simulering av riskområden



David Wennberg
Civilingenjörsprogrammet i Teknisk Fysik
Lunds Tekniska Högskola

Förord

Examensarbetet *Beräkningar av riskområden i Matlab* utfördes hösten 2008 på SAAB Bofors Dynamics i Eskilstuna. Syftet med examensarbetet var att ta fram ett nytt verktyg för beräkning av riskområden.

Tack till:

Håkan Näsberg (SBD):	Handledare på SAAB Bofors Dynamics
Per Lidström (LTH):	Handledare på Mekanik institutionen på Lunds Tekniska Högskola.
Olle Karlsson (SBD):	För hjälp med beräkning av projektilbanor och generell MATLAB-programmeringen.

Ytterligare tack till hela sektionen på SAAB Bofors Dynamics för stödet under utförandet av examensarbetet.

Innehållsförteckning

1 Inledning

Försvarsmaktens säkerhetsinstruktion, SäkI, definierar ett riskområde som ”… det område där skador kan uppstå vid viss verksamhet…” (*SäkI G, 2008*, s. 49 [1]). SäkI definierar sedan olika verksamheter som kan skapa ett riskområde, till exempel: skjutning, sprängning, buller, arbetande motorredskap med mera. Riskområdet som analyseras i detta examensarbete är riskområdet som uppstår vid skjutning.

På SAAB Bofors Dynamics beräknas riskområden med ett DOS-baserat program som utifrån ett tiotal geometriska inparametrar, liknande de definierade i försvarsmaktens säkerhetsinstruktioner (*SäkI G, 2008*, kap 4: Riskområden [1]), skapar enkla riskområden bestående av avgränsningslinjerna för riskområdet utanför vilka risken är försumbar. Vid skapandet av riskområden behöver man därför bestämma vad som är försumbar risk. Detta varierar dessutom beroende på om man undersöker risk för första, andra eller tredjeman.

Uppgiften i detta examensarbete var att ta fram ett nytt verktyg för beräkning av riskområden vid skjutning. De nya riskområdena skulle vara framtagna via simuleringar på så sätt att sannolikheterna för riskerna som uppstår vid skjutning kan skådas direkt i riskområdet varav behovet att definiera vad som kan ses som försumbar risk innan riskområdet skapas försvinner. Därefter skulle de simulerade områdena kunna jämföras med riskområdena definierade i SäkI.

Med hänsyn till detta strukturerades examensarbetet i två delar. Första delen bestod i att ta fram program för beräkning av SäkI:s riskområden. Detta program skulle fungera på liknande sätt som tidigare DOS-program som använts på SAAB Bofors Dynamics men vara skrivet i beräkningsprogrammet MATLAB [8]. Andra delen bestod av modellering och simulering av skjutfall i MATLAB. Resultatet av simuleringarna skulle bygga upp de nya riskområdena. Detta görs genom att Monte Carlo-simulera skjutfall.

2 Metodik

För att förtydliga att vissa parametrar är matriser eller vektorer i denna rapport betecknas dessa, vid behov, enligt nedan.

Matris A	
i text	i ekvationer
A	$\overline{A}$

2.1 SäkI:s riskområden i MATLAB

I Försvarsmaktens säkerhetsinstruktioner, SäkI, definieras ett 20-tal olika parametrar för beräkning av riskområden (se bilaga 1: *Struktursamling*, struktur *RiscData*). Med dessa parametrar byggs det totala riskområdet utifrån det givna riskfallet. SäkI definierar tre olika riskfall enligt nedan (*SäkI G 2008*, s. 56, *Riskfall* [1]):

		Riskfall vid skjutning mot*		
Kaliber	Ammunition	Vatten eller is	Stenbunden mark, metall och betong	Övrig mark**
<20mm	Med/utan tändrör	A	B	C
≥20mm	Utan tändrör	A	A***	B****
	Med tändrör	A	B****	C

* Är marken snötäckt räknar man med riskfall för underliggande mark
** Är marken tjälad tillämpas riskfall enligt kolumnen för ”stenbunden mark, metall, betong”
*** För projektiler för vilken hastigheten i nedslagspunkten understiger 400m/s, får riskfall B användas vid stenbunden eller tjälad mark
**** enligt not ***, dock får riskfall C användas

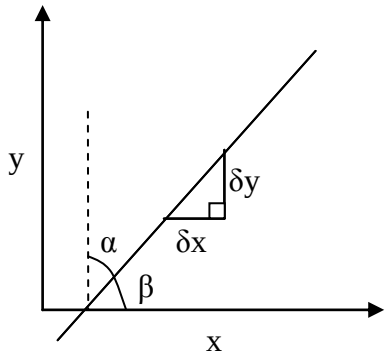
Riskområdena som presenteras i SäkI är konstruerade för att uppritas för hand med papper och penna. För att modellera dessa riskområden i MATLAB har skytten placerats i origo med sikte på ett givet målområde. Med inparametrarna skapas sedan geometriska områden begränsade av räta linjer och cirkelsektorer. Riskområdet begränsas av skärningspunkterna mellan dessa linjer.

Analys av delat riskområde (se *SäkI G 2008*, s 62, *Mynningsavstånd* [1]) ingår inte i SäkI:s riskområden i MATLAB.

SäkI innehållsförteckning

2.1.1 Teori

SäkI:s riskområden är uppbyggda med geometriska parametrar varför ekvationer för räta linjen samt cirkelsektorer kommer väl till användning.



Figur 1.1, räta linjens ekvation

Räta linjens ekvation:

T1.e1

$$y = k \cdot x + c$$

$$k = \frac{dy}{dx} = \tan(\beta) = \tan(\frac{\pi}{2} - \alpha)$$

Ansatsen till k-värdet i T1.e1 används ständigt i framtagningen av SäkI riskområden och klargörs i figur 1.1.

En annan viktig formel är den för skärningspunkten [x_s y_s] mellan två linjer i samma plan. Utryck T1.e2 förtydligar ekvationen för två olika linjer och T1.e3 ger skärningspunkten [x_s y_s] mellan dessa linjer. I uttrycket för y_s kan *i* väljas till 1 eller 2.

T1.e2

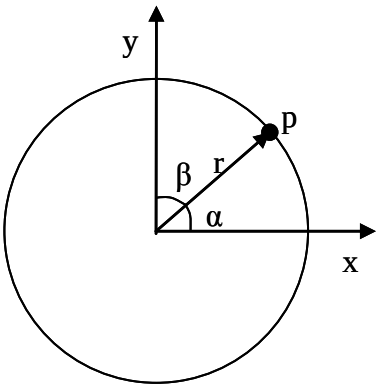
$$y_1 = k_1 \cdot x + c_1$$

$$y_2 = k_2 \cdot x + c_2$$

T1.e3

$$x_s = \frac{c_1 - c_2}{k_2 - k_1}$$

$$y_s = x_s \cdot k_i + c_i$$



Figur 1.2, cirkel med raide = *r*

I figur 1.2 återfinns en cirkel med radie lika med *r*. På cirkelbågen är punkten *p* utmarkerad. Ekvationen för denna punkt ges av:

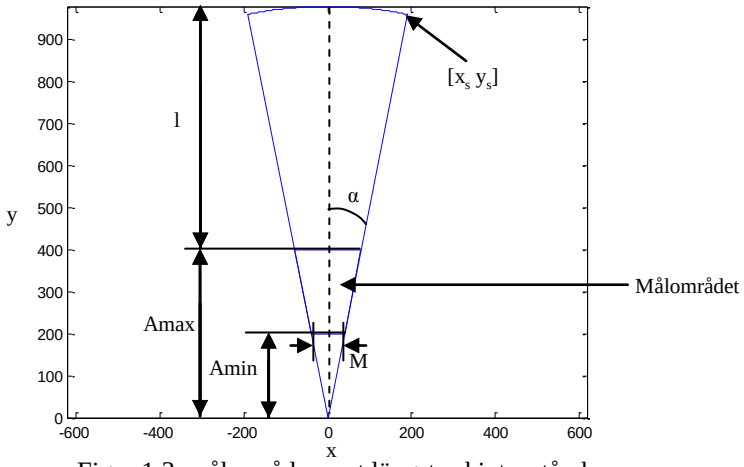
T1.e4

$$x_p = r \cdot \cos(\alpha) = r \cdot \sin(\beta)$$

$$y_p = r \cdot \sin(\alpha) = r \cdot \cos(\beta)$$

Dessa ekvationer ger grunden för beräkning av SäkI:s riskområden i MATLAB.

2.1.1.1 Mål- och skjutområde med sidspridning



Figur 1.3, målområde samt längsta skjutavstånd

Målområdet i figure 1.3 ovan beräknas utifrån målområdets främre bredd *M*, eller direkt med vinkeln i sida, α, samt *Amax* och *Amin*. Avstånd bortom målområdet, *l*, beräknas utifrån riskfallet som ska analyseras. Cirkelsektorn i figur 1.3 består av två räta linjer enligt ekvation T1.e1:

T1.e5

$$y_1 = k_\alpha \cdot x$$

$$y_2 = -k_\alpha \cdot x$$

$$k_\alpha = \tan(\frac{\pi}{2} - \alpha)$$

$$\alpha = \arctan(\frac{M}{2 \cdot A\max})$$

Linjerna börjar i origo och avslutats då de skär cirkeln med radie $h = l + Amax$. I figur 1.3 är en av dessa skärningspunkter, $[x_s\ y_s]$, utmarkerad i första kvadranten. Denna punkt ges av T1.e4 enligt:

T1.e6

$$x_s = h \cdot \sin(\alpha)$$

$$y_s = h \cdot \cos(\alpha)$$

motsvarande punkt på negativa *x*-axeln fås genom att spegla koordinaten i linjen $x = 0$. Eftersom riskområdena är symetriska kring linjen $x = 0$ kommer endast skärningspunkter i första och fjärde kvadranten redovisas i följande beräkningar om inte annat anges. Cirkelbågens koordinater, $[x_c\ y_c]$, ges av:

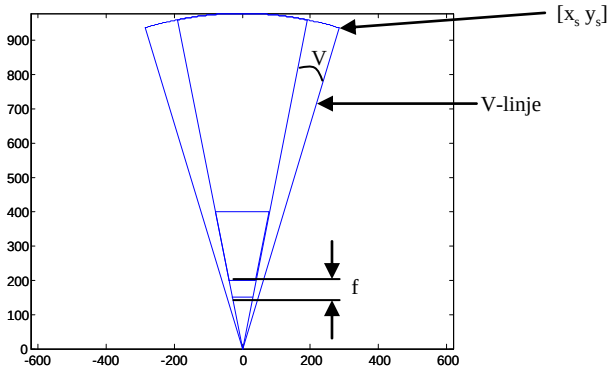
T1.e7

$$x_c = h \cdot \sin(\varphi)$$

$$y_c = h \cdot \cos(\varphi)$$

$$-\alpha < \varphi < \alpha$$

Nästa steg i skapandet av SäkI:s riskområden är att lägga till riskavståndet för direktträff hitom målområdet, *f*, samt riskvinkel för sidspridning, *V* (*SäkI G 2008, s.54-55* [1]), se figur 1.4.



Figur 1.4, riskavstånd för direktträff hitom skjutområdet samt riskvinkel för sidspridning

I figur 1.4 har *V*-linjen definierats för att underlätta i beskrivningen av resterande delar av riskområdet.

V-linjen börjar i origo och sträcker sig till skärningen av cirkeln med radie = *h*. skärningspunkten ges av T1.e4 enligt:

T1.e8

$$x_s = h \cdot \sin(\alpha + V)$$
$$y_s = h \cdot \cos(\alpha + V)$$

V-linjens ekvation beräknas med T1.e1 enligt:

T1.e9

$$y = k_V \cdot x$$
$$k_V = \tan(\frac{\pi}{2} - \alpha - V)$$

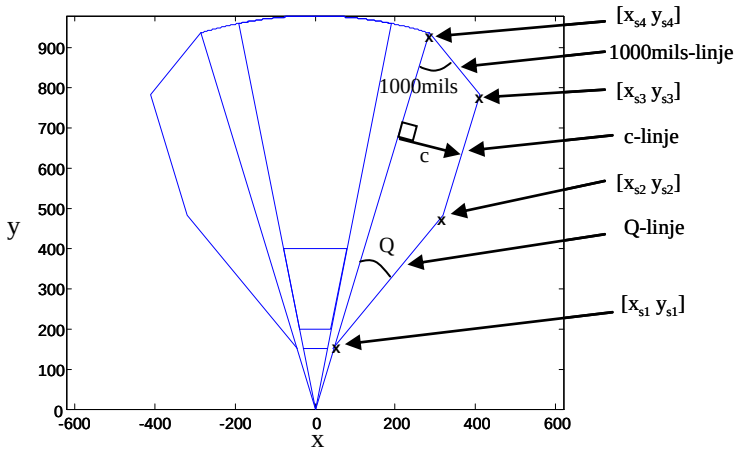
Cirkelsektorn överst i figur 1.3 förlängs åt sidorna för att täcka in riskvinkeln för sidspridning enligt:

T1.e10

$$x_c = h \cdot \sin(\varphi)$$
$$y_c = h \cdot \cos(\varphi)$$
$$-(\alpha + V) < \varphi < \alpha + V$$

Härefter tillkommer olika områden beroende på vilket scenario som studeras.

2.1.1.2 Riskvinklar och avstånd på grund utav studs



Figur 1.5, Riskvinklar och avstånd på grund av studs

Riskområdena som uppkommer på grund utav studs visas i figur 1.5. Området börjar med riskvinkeln för studs *Q* som utgår från skärningspunkten mellan *V*-linjen och förlängningen av linjen för direktträff hitom skjutområdet, se figur 1.4, i koordinat $[x_{s1}\ y_{s1}]$ (*SäkI G 2008, s. 57* [1]).

Q-linjen beräknas med ekvation T1.e1 och blir:

T1.e11

$$y = k_Q \cdot x + c_Q$$

$$k_Q = \tan(\frac{\pi}{2} - \alpha - V - Q)$$

c_Q fås genom att ansätta *Q*-linjen lika med *V*-linjen (se T1.e9) i *y = Amin-f*.

T1.e12

$$y = k_V \cdot x = (A\min - f) \Rightarrow x = \frac{A\min - f}{k_V} (1)$$

$$y = k_Q \cdot x + c_Q = (A\min - f) \Rightarrow (1) \Rightarrow$$

$$c_Q = (1 - \frac{k_Q}{k_V}) \cdot (A\min - f)$$

Koordinaten $[x_{s1}\ y_{s1}]$ i figur 1.5 ges av:

T1.e13

$$x_{s1} = \frac{A\min - f}{k_V}$$

$$y_{s1} = A\min - f$$

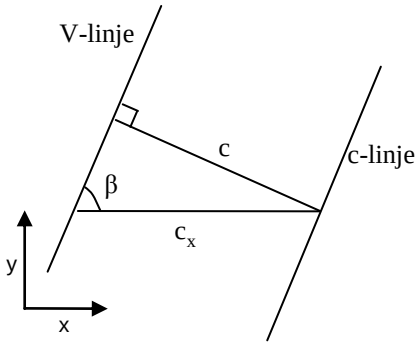
Q-linjen fortsätter sedan upp till skärning av *c*-linjen enligt figur 1.5. Riskavståndet i sida vid studs, *c*, är ett mått som tar hänsyn till energiförluster som uppstår vid studs

och begränsar avståndet projektiler kan studsa i sidled (*SäkI G 2008, s. 58* [1]). *C*-linjen går parallellt med *V*-linjen (se T1.e9) och har därför samma lutning som *V*-linjen i räta linjens ekvation enligt:

T1.e14

$$y = k_c \cdot x + c_c = k_V \cdot x + c_c$$

C-linjens konstant c_c fås genom att beräkna c_x ur geometrin i figur 1.6 och lösa ut c_c ur ekvation T1.e14 i $y = 0$, $x = c_x$.



Figur 1.6, c-linjen

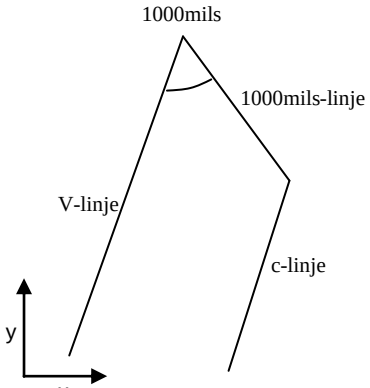
T1.e15

$$c_c = -k_V \cdot c_x$$

$$c_x = \frac{c}{\sin(\beta)}$$

$$\beta = \frac{\pi}{2} - \alpha - V$$

Skärningspunkten [x_{s2} y_{s2}], i figur 1.5, mellan *Q*-linjen och *c*-linjen fås med ekvation T1.e3 och blir:



Figur 1.7, 1000mils linjen

T1.e16

$$x_{s2} = \frac{c_c - c_Q}{k_Q - k_c}$$

$$y_{s2} = x_{s2} \cdot k_c + c_c$$

Där värdena för de olika parametrarna i T1.e16 ges av ekvation T1.e11, T1.e12, T1.e14 och T1.e15.

C-linjen bryts till slut av *1000mils*-linjen (6400mils motsvarar 2π rad). Denna linje uppkommer på grund av att projektiler som studsat utanför *V* har förlorat energi vilket gör att *h*, längsta skjutavståndet, reduceras (*SäkI G 2008, s. 60* [1]). *1000mils*-linjen har en lutning som utifrån *V*-linjen skiljer sig med π-1000mils, se figur 1.7. Detta resulterar i den räta linjen:

T1.e17

$$y = k_{1000mils} \cdot x + c_{1000mils}$$

$$k_{1000mils} = \tan\left(\frac{\pi}{2} - V - \alpha - \left(\pi - \frac{1000}{6400} \cdot 2\pi\right)\right) =$$

$$= \tan\left(\frac{10}{32} \cdot \pi - V - \alpha - \frac{\pi}{2}\right)$$

Genom att använda den sista skärningspunkten i figur 1.5, [x_{s4} y_{s4}], som är känd utifrån geometrin och ges av:

T1.e18

$$x_{s4} = h \cdot \sin(\alpha + V)$$
$$y_{s4} = h \cdot \cos(\alpha + V)$$

och sätta in den i *1000mils*-linjens ekvation fås konstanten $c_{1000mils}$ till:

T1.e19

$$c_{1000mils} = y_{s4} - k_{1000mils} \cdot x_{s4}$$

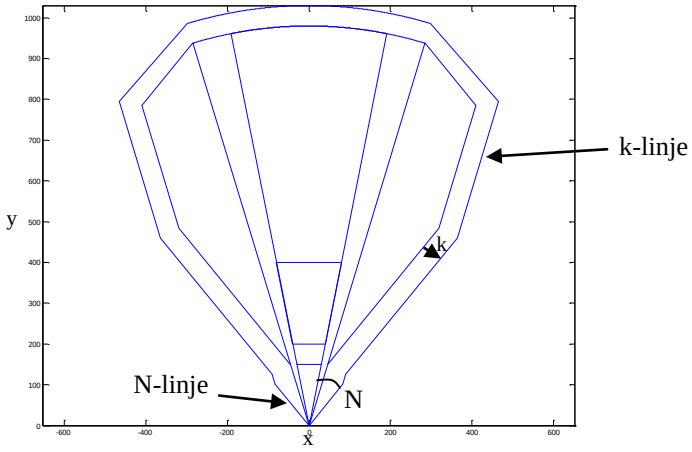
Den återstående skärningspunkten mellan *1000mils*-linjen och *c*-linjen fås genom insättning av de framtagna värdena i ekvation T1.e3 enligt:

T1.e20

$$x_{s3} = \frac{c_c - c_{1000mils}}{k_{1000mils} - k_c}$$
$$y_{s3} = x_{s2} \cdot k_c + c_c$$

2.1.1.3 Riskavstånd på grund utav splitter

Riskavstånd *k* är det avstånd inom vilket det finns risk för splitter från eventuell krevad/brisad eller splitter från terräng (*SäkI G 2008, s. 62-64* [1]). *k* är en vinkelrät förflyttning utåt utanför det nuvarande riskområdet, se figur 1.8.



Figur 1.8, riskavstånd på grund utav splitter

Detta område skapas genom att använda de redan kända linjerna och beräkna nya konstanter i respektive linjes ekvation. Eftersom *k* är en vinkelrät förflyttning utanför det nuvarande området blir *k*-linjen och den innanliggande linjen parallella. Den nya konstanten i *k*-linjen ges av:

T1.e21

$$c_{ki} = c_i - \frac{sign(k_i) \cdot k}{\cos(\arctan(k_i))}$$

I ekvation T1.e21 representerar c_{ki} konstanten för *k*-linjen som går parallellt med godtycklig linje *i*, k_i är lutningen på motsvarande linje och *k* är riskavståndet för splitter.

Skärningspunkter mellan de olika *k*-linjerna ges av ekvation T1.e3 enligt:

T1.e22

$$x_{si} = \frac{c_{ki} - c_i}{k_i - k_{ki}}$$

$$y_{si} = x_{si} \cdot k_{ki} + c_{ki}$$

Framför mynningen uppkommer ett riskområde som beror på mynningsvinkeln *N*, se figur 1.8. Storleken på *N* beror främst på partiklar från krutförbränningen, mynningsflamman och mynningstryckets utbredning (*SäkI G 2008, s. 61* [1]). Denna vinkel utgår från α-vinkeln och *N*-linjens ekvation blir i enlighet med T1.e1 således:

T1.e23

$$y = x \cdot k_N = x \cdot \tan(\frac{\pi}{2} - \alpha - N)$$

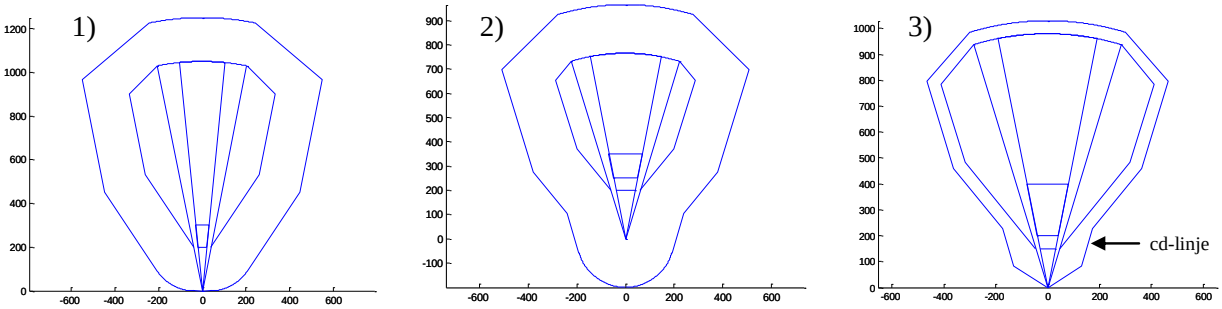
N-linjen avslutas i skärning med *k*-linjen enligt figur 1.8.

Det förekommer en del specialfall kring skytten vid beräkning av riskområdet ifrån splitter.

Exempel på specialfall

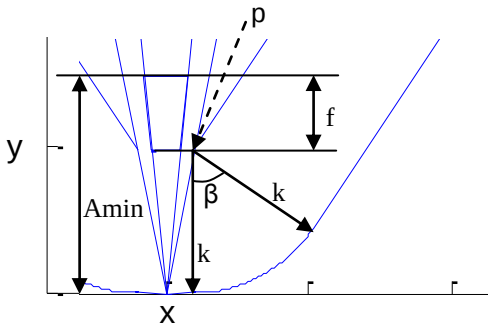
1) *Amin = f + k* (*SäkI G 2008, s. 63* [1]).
2) Ansatt *k1* värde
3) Studs av drivspegel hos underkalibrig ammunition (*SäkI G 2008, s. 64* [1]).

Specialfallen ovan visas i figur 1.9 nedan.



Figur 1.9, Special fall av riskområden, 1) Amin = f+k, 2) ansatt k1 värde, 3) studs av drivspegel

Dessa fall undersöks varje gång ett riskområde ska beräknas för att finna de korrekta områdena.



Figur 1.10, riskområdet sträcker sig tillbaka till skytten

Fall 1) Riskavståndet för splitter sträcker sig tillbaka till skytten

Vid denna händelse skapas två cirkelsektorer, en i första kvadranten och en i andra. Figur 1.10 illustrerar cirkelsektorn i första kvadranten. Centrum för cirkelsektorn i figur 1.10 ges av punkten $p = \left[(A\min - f)\cdot\tan(V+\alpha), A\min - f\right]$. Cirkelsektorn börjar i koordinaten $\left[(A\min - f)\cdot\tan(V+\alpha), 0\right]$ och fortsätter sedan i cirkelbågen i figur 1.10 enligt:

T1.e24

$$x_c = (A\min - f)\cdot\tan(V+\alpha) + k\cdot\sin(\varphi)$$

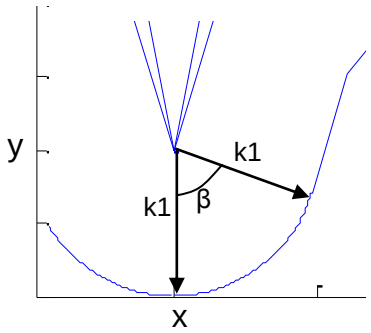
$$y_c = k\cdot(1-\cos(\varphi))$$

$$0 \le \varphi \le \beta$$

Vinkeln β, i figur 1.10 samt ekvation T1.e24, bestäms utifrån när cirkelbågen skär *k*-linjen parallell med *Q*-linjen. Cirkelsektorn i andra kvadranten fås genom att spegla cirkelsektorn i första kvadranten i linjen *x = 0*. De två cirkelsektorerna sammanbinds i origo med den räta linjen *y = 0*.

Fall 2) Parameter *k1* är ansatt och riskområdet för splitter börjar vid vapenmynningen

Cirkelsektorn som sträcker sig bakom mynningen i fjärde kvadranten visas i figur 1.11 och ges av ekvation T1.e25 nedan.



Figur 1.11, ansatt *k1*-värde

T1.e25

$$x_c = k1 \cdot \sin(\varphi)$$

$$y_c = -k1 \cdot \cos(\varphi)$$

$$0 \le \varphi \le \beta$$

Vinkeln β i figur 1.11, samt ekvation T1.e25, fås genom att finna skärningen mellan cirkelbågen och *k*-linjen som går parallellt med *V*-linjen eller *k*-linjen som går parallellt med *Q*-linjen beroende på vilken skärningspunkt som inträffar först då vinkeln φ i ekvation T1.e25 ökas från 0. Detta resulterar i att ta reda på vilken av följande residualer som först går mot noll.

T1.e26

$$res1 = -k1 \cdot \cos(\beta) - k_{kV} \cdot k1 \cdot \sin(\beta) - c_{kV}$$

$$res2 = -k1 \cdot \cos(\beta) - k_{kQ} \cdot k1 \cdot \sin(\beta) - c_{kQ}$$

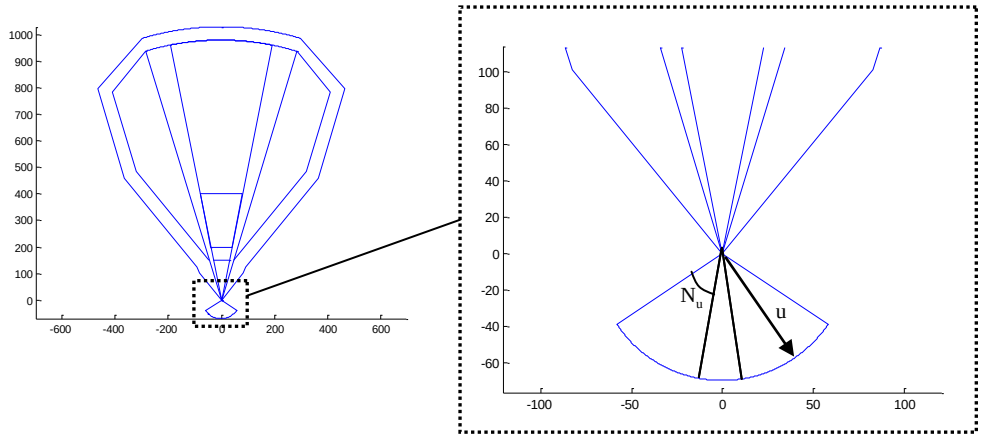
Här är, som tidigare, k_{kV} och c_{kV} lutning respektive konstant för *k*-linjen parallell med *V*-linjen och på motsvarande sätt är k_{kQ} och c_{kQ} lutning respektive konstant för k-linjen parallell med *Q*-linjen. Dessa residualer motsvarar skärningarna mellan cirkelbågen nedtill i figur 1.11 samt *k*-linjerna parallella med *V*-linjen respektive *Q*-linjen.

Fall 3) Parameter *cd* är ansatt och det finns risk för studs av drivspegel

I detta fall blir riskområdet för splitter förflyttat avstånd *cd* utifrån *V*-linjen istället för avstånd *k*. *cd*-linjen skärs av *N*-linjen och återstående *k*-linje, se figur 1.9 3).

2.1.1.4 Riskavstånd bakom vapnet

Riskavståndet bakom vapnet är det riskavstånd som uppstår vid skjutning med vapen som har gasutströmning bakåt (*SäkI G 2008, s. 64-65* [1]).
Figur 1.12 illustrerar hur riskområdet bakom vapnet är definierat. Vinkeln N_u utgår



Figur 1.12, riskavstånd bakom vapnet

från α-linjens förlängning bakom vapnet.
Cirkelsektorn blir således:

T1.e27

$$x_c = u \cdot \sin(\beta)$$

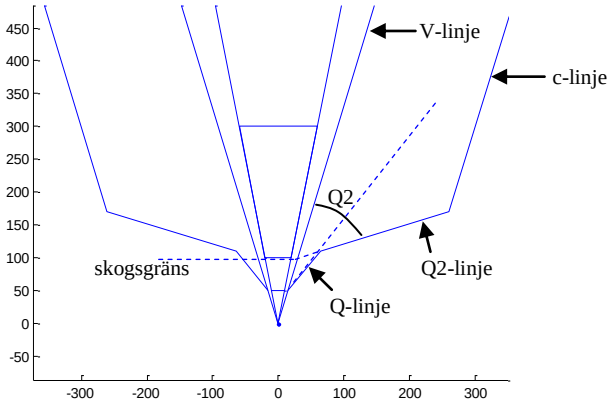
$$y_c = -u \cdot \cos(\beta)$$

$$-(N_u + \alpha) < \beta < N_u + \alpha$$

2.1.1.5 Riskområden vid skjutning i och mot skog

Vid skjutning mot skog tillkommer studsvinkeln *Q2* (*SäkI Ehv/Pv 2008, s. 132* [2]).

Q2 utgår från *V*-linjen (se T1.e9) och skogsgränsen enligt figur 1.13, men börjar först verka efter skärning med *Q*-linjen. *Q2*-linjen fortsätter sedan till skärning med *c*-linjen enligt figur 1.13.



Figur 1.13, skjutning mot skog

Q2-linjens ekvation beräknas med ekvation T1.e1 enligt:

T1.e28

$$y = k_{Q2} \cdot x + c_{Q2}$$

$$k_{Q2} = \tan(\frac{\pi}{2} - V - Q2)$$

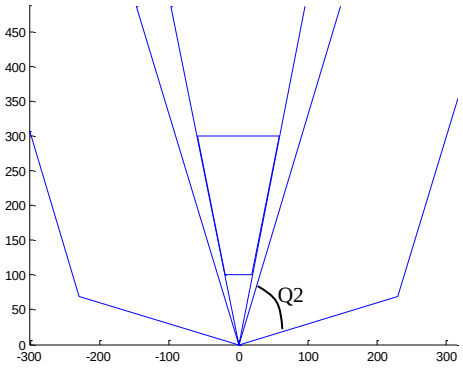
C_{Q2} fås genom att ansätta *V*-linjen och *Q2*-linjen lika i *y* = *skogsgränsen*. Detta resulterar i:

T1.e29

$$c_{Q2} = skogsgräns \cdot (1 - \frac{k_{Q2}}{k_V})$$

Skogsgräns är avståndet till skogen i meter på linjen *x* = 0 ifrån origo.

Vid skjutning i skog (*SäkI Ehv/Pv 2008, s. 132* [2]) utgår *Q2*-linjen ifrån origo, skogsgränsen ligger då på *y* = 0, se figur 1.14.

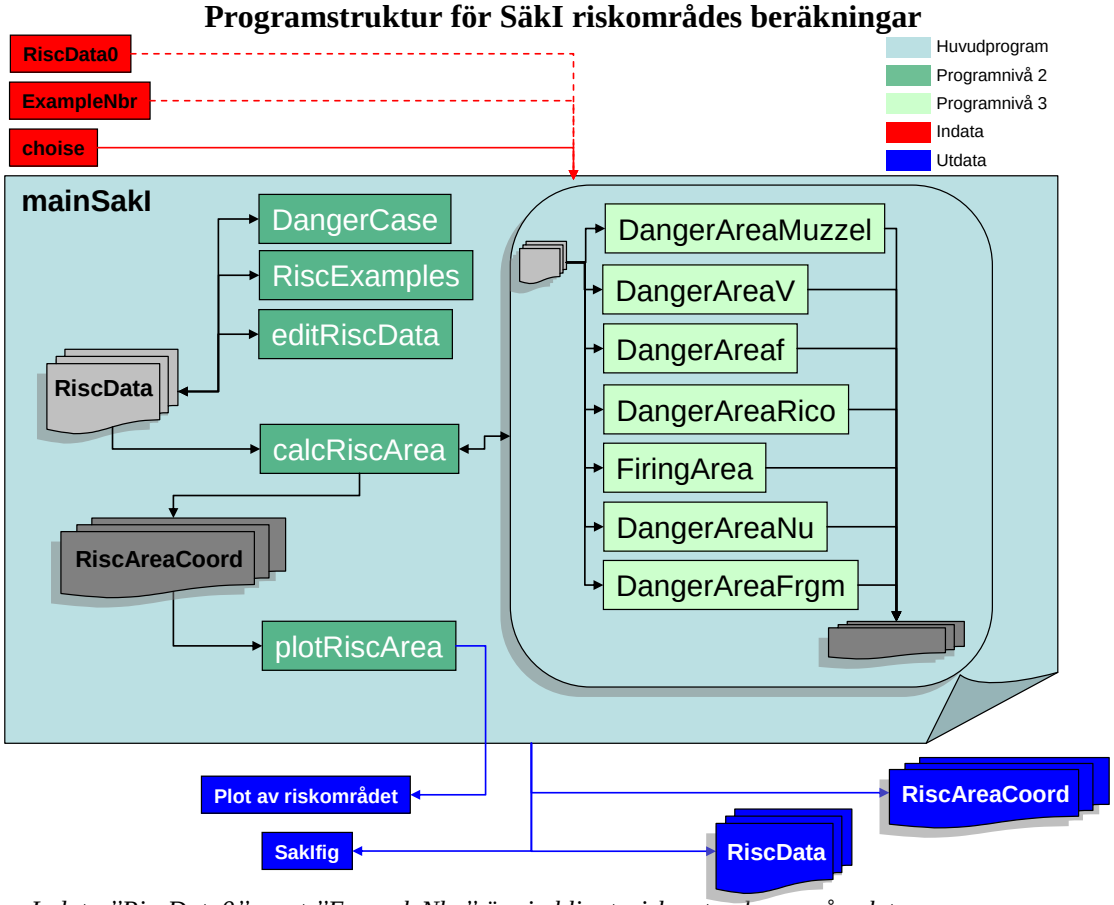


Figur 1.14, skjutning i skog

2.1.2 SäkI:s riskområden i MATLAB

MATLAB programmet för beräkning av SäkI riskområden arbetar utifrån definitionerna i försvarsmaktens säkerhetsinstruktioner för konstruktion av riskområden (*SäkI G 2008*, kapitel 4 [1], *SäkI Ehv/Pv 2008*, kapitel 8 [2]) samt ekvationer och samband definierade i teori avsnitt 2.1.1. Beräkningarna ger riskområdets utbredning i ett plan och ingen information om höjdspridning. SäkI har definierat riskavstånd i höjd som funktion av en studsfaktor, *s* (*SäkI G 2008, sida 65* [1]), detta är dock inget som illustreras i deras riskområden eller de riskområden som beräknas här.

Nedan visas ett schema över hur programmet för beräkning av SäkI riskområden arbetar.



Indata "RiscData0" samt "ExampleNbr" är ej obligatoriska utan beror på valet av "choise", därav de streckade röda linjerna.

Programmet arbetar med en struktur, *RiscData*, som innehåller samtliga variabler för beskrivning av riskområdet. Denna struktur skickas mellan olika funktioner för beräkning av koordinater som begränsar riskområdets olika områden enligt teorin i avsnitt 2.1.1. Koordinaterna sparas i matriser som i sin tur sparas i strukturen *RiscAreaCoord*. Riskområdet plottas genom att koordinaterna i *RiscAreaCoord* sammanbinds med MATLAB-funktionen *plot* [8]. Strukturerna återfinns i bilaga 1.

För mer information om funktionerna i de olika programnivåerna se avsnitt 2.1.4: *Funktionsbeskrivningar*. För information om in- och utdata till huvudprogrammet *mainSakI* se avsnitt 2.1.4.14: *mainSakI*.

2.1.3 SäkI-funktioner

Följande funktioner används vid beräkning av SäkI:s riskområden i MATLAB.

Tabell 1, funktioner för beräkning av SäkI:s riskområden i MATLAB

Benämning	Beskrivning
FiringArea	Beräknar skjutområde samt målområde
DangerAreaMuzzel	Beräknar riskområde framför mynning
DangerAreaRico	Beräknar riskområde som uppkommer från ricochet
DangerAreaV	Beräknar riskområde för sidspridning
DangerAreaNu	Beräknar riskområde bakom vapnet
DangerAreaf	Beräknar koordinaterna för markering av riskavståndet
DangerAreaFrgm	Beräknar riskområde för splitter
DangerCase	Beräknar parametrarna c och l utifrån riskfall
calcRiscArea	Beräknar koordinater som begränsar riskområdet
editRiscData	GUI
plotRiscArea	Ritar upp riskområdet med hjälp av koordinater
RiscExamples	Samling exempel med indata
mainSakI	Styrprogram

2.1.4 Funktionsbeskrivningar

Nedan beskrivs de funktioner som används vid beräkning av SäkI:s riskområden i MATLAB.

2.1.4.1 calcRiscArea

DangerAreaMuzzel Funktionen skapar strukturen *RiscAreaCoord* som innehåller koordinatmatriserna som begränsar riskområdets olika delar. *calcRiscArea* använder sig av funktionerna i tabell 1, avsnitt 2.1.3, för att beräkna de olika matriserna. I tabellen nedan är samtliga matriser och deras respektive funktioner listade.

Matris	**Funktion**
FA	FiringArea
TA	FiringArea
DngA_V	DangerAreaV
DngA_f	DangerAreaf
DngA_Nu	DangerAreaNu
DngA_k	DangerAreaFrgm
DngA_rN	DangerAreaMuzzel
DngA_Qc	DangerAreaRico

Matriserna i *RiscAreaCoord* är konstruerade för plottning med MATLAB-funktionen *plot* [8]. Matriserna består av två kolonner där första kolonnen innehåller *x*-koordinater och andra innehåller *y*-koordinater. Vissa av matriserna är endast gjorda för halva riskområdet (positiva *x*) och får därför speglas i linjen $y = 0$ för plottningen av totala riskområdet. Detta görs automatiskt av funktionen *plotRiscArea* (se avsnitt 2.1.4.4).

Syntax:

RiscAreaCoord = calcRiscArea(RiscData)

Indata:

RiscData: Struktur med parametrar som beskriver riskområdet, se bilaga 1.

Utdata:

RiscAreaCoord: Struktur med koordinatmatriser som begränsar riskområdet, se bilaga 1.

2.1.4.2 DangerAreaf

Beräknar koordinaterna för *f*-linjen som visar riskavståndet för direktträff hitom skjutområdet (se figur 1.4, avsnitt 2.1.1.1).

Linjen består av två koordinater i *xy*-planet som ges av:

2.1.4.2.e1

$$DngA\_f = \begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \end{bmatrix} = \begin{bmatrix} -(A\min - f)\cdot\tan(\alpha) & A\min - f \\ (A\min - f)\cdot\tan(\alpha) & A\min - f \end{bmatrix}$$

Syntax:

DngA_f = DangerAreaf(RiscData)

Indata:

RiscData: Struktur med parametrar som beskriver riskområdet, se bilaga 1.

Utdata:

DngA_f: Matris av storlek 2x2 med koordinater för *f*-linjen.

2.1.4.3 editRiscData

GUI (Graphical User Interface) för inmatning av parametrarna i strukturen *RiscData*, se bilaga 1: *Struktursamling*.

Syntax:

[RiscData, cancel] = editRiscData

Utdata:

RiscData:	Struktur med parametrar som beskriver riskområdet, se bilaga 1.
cancel:	Boolean, *cancel* = *true* om GUIn avslutas annars *false*.

2.1.4.4 plotRiscArea

Plottar riskområdet i MATLAB-fönster med hjälp av koordinaterna i *RiscAreaCoord* samt MATLAB-funktionen *plot* [8].

De matriser som endast är beräknade för halva riskområdet speglas i linjen $x = 0$ för plot av totala riskområdet.

Syntax:

plotRiscArea(RiscAreaCoord)

Indata:

RiscAreaCoord: Struktur med koordinatmatriser för de olika områdena av riskområdet.

Utdata:

Plot av riskområdet.

2.1.4.5 FiringArea

Beräknar koordinaterna för skjutområdet, *FA*, och målområdet, *TA*, utifrån indata *alpha* (α), *h*, *Amin* och *Amax* enligt avsnitt 2.1.1.1 *Mål- och skjutområde med sidspridning*.

Matrisen *TA* ges av:

2.1.4.5.e1

$$TA = \begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \\ x_3 & y_3 \\ x_4 & y_4 \\ x_5 & y_5 \end{bmatrix} = \begin{bmatrix} -A\min \cdot \tan(\alpha) & A\min \\ -A\max \cdot \tan(\alpha) & A\max \\ A\max \cdot \tan(\alpha) & A\max \\ A\min \cdot \tan(\alpha) & A\min \\ -A\min \cdot \tan(\alpha) & A\min \end{bmatrix}$$

Matrisen *FA* ges av:

2.1.4.5.e2

$$FA = \begin{bmatrix} x_1 & y_1 \\ . & . \\ . & . \\ . & . \\ x_n & y_n \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ h \cdot \sin(\alpha) & h \cdot \cos(\alpha) \\ x_3 & y_3 \\ . & . \\ . & . \\ x_{n-2} & y_{n-2} \\ -h \cdot \sin(\alpha) & h \cdot \cos(\alpha) \\ 0 & 0 \end{bmatrix}$$

Koordinaterna x_3 till x_{n-2} samt y_3 till y_{n-2} ges av ekvation T1.e7 i avsnitt 2.1.1.1. Dessa utgör koordinaterna för längsta skjutavståndet.

Första och sista koordinaten är lika i båda matriserna för att avsluta målområdet i samma punkt som den började.

Syntax:

[FA, TA] = FiringArea(RiscData)

Indata:

RiscData: Struktur med parametrar som beskriver riskområdet, se bilaga 1.

Utdata:

FA: Matris av storlek nx2 där *n* blir godtyckligt stort beroende på indata. Matrisen innehåller koordinaterna som avgränsar

skjutområdet. Första kolonen representerar *x* koordinater och andra *y* koordinater.

TA: Matris av storlek 5x2. Matrisen innehåller koordinater som avgränsar målområdet. Första kolonen representerar *x* koordinater och andra *y* koordinater.

2.1.4.6 DangerAreaV

Beräknar koordinaterna, *DngA_V*, för riskområdet som uppstår på grund utav sidspridning enligt avsnitt 2.1.1.1 *Mål- och skjutområde med sidspridning*

2.1.4.6.e1

$$DngA\_V = \begin{bmatrix} x_1 & y_1 \\ \cdot & \cdot \\ \cdot & \cdot \\ \cdot & \cdot \\ x_n & y_n \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ h \cdot \sin(\alpha + V) & h \cdot \cos(\alpha + V) \\ x_3 & y_3 \\ \cdot & \cdot \\ \cdot & \cdot \\ x_{n-2} & y_{n-2} \\ -h \cdot \sin(\alpha + V) & h \cdot \cos(\alpha + V) \\ 0 & 0 \end{bmatrix}$$

Koordinaterna x_3 till x_{n-2} samt y_3 till y_{n-2} ges av ekvation T1.e10 i avsnitt 2.1.1.1.

Syntax:

DngA_V = DangerAreaV(RiscData)

Indata:

RiscData: Struktur med parametrar som beskriver riskområdet, se bilaga 1.

Utdata:

DngA_V: Matris av storlek *n*x2 där *n* blir godtyckligt stort beroende på indata. Matrisen innehåller koordinaterna som avgränsar riskområdet för sidspridning. Första kolonen representerar *x* koordinater och andra *y* koordinater.

2.1.4.7 DangerAreaRico

Beräknar koordinaterna, *DngA_Qc*, för riskområdet som uppstår på grund av ricochetter vid eventuell studs enligt avsnitt 2.1.1.2 *Riskvinklar och avstånd på grund utav studs* samt avsnitt 2.1.1.5 *Riskområden vid skjutning i och mot skog*.

DngA_Qc består av skärningspunkterna [x_{si} y_{si}], i = 1 till 3, 4 eller 5 beroende på inparametrarna i strukturen *RiscData* (se bilaga 1: *Struktursamling*).

Följande skärningspunkter beräknas (*OBS: endast positiva x*):

P_{VQ}:	*V*-linjens skärning med *Q*-linjen
P_{VQ2}:	*V*-linjens skärning med *Q2*-linjen
P_{QQ2}:	*Q*-linjens skärning med *Q2*-linjen
P_{Q2c}:	*Q2*-linjens skärning med *c*-linjen
P_{Qc}:	*Q*-linjens skärning med *c*-linjen
P_{redc}:	*1000mils*-linjen med *c*-linjen
P_{redQ}:	*1000mils*-linjen med *Q*-linjen
P_{redQ2}:	*1000mils*-linjen med *Q2*-linjen

Linjerna ges enligt teorin (avsnitt 2.1.1.1, 2.1.1.2 samt 2.1.1.5) med följande ekvationer

Q-linjen:	T1.e11 samt T1.e12
V-linjen:	T1.e9
Q2-linjen:	T1.e28 samt T1.e29
c-linjen:	T1.e14 samt T1.e15
1000mils-linjen:	T1.e17, T1.e18 samt T1.e19

Skärningspunkterna ges sedan av ekvation T1.e3.

Vilka skärningspunkter som ska ingå i *DngA_Qc* avgörs av punkternas *x*-koordinater. Vid tillräckligt stora *c*-värden kommer till exempel *1000mils*-linjen skära *Q*-linjen före *c*-linjen vilket resulterar i att skärningspunkt P_{redQ}:s *x*-koordinat är mindre än P_{redc}:s *x*-koordinat och skärningspunkten P_{redc} ingår då inte i *DngA_Qc*, jämför figur 1.5. Vid skjutning i skog hamnar skärningspunkten P_{VQ2} i origo och skärningspunkten P_{VQ} ingår då inte i *DngA_Qc*.

Riskområdet avslutas alltid i skärningen mellan *1000mils*-linjen och *V*-linjen, koordinat [x_{end} y_{end}] som ges av ekvation T1.e18.

2.1.4.7.e1

$$DngA\_Qc = \begin{bmatrix} x_{s1} & y_{s1} \\ . & . \\ . & . \\ . & . \\ x_{sn} & y_{sn} \\ x_{end} & y_{end} \end{bmatrix}, \quad n = 3,\ 4,\ 5$$

Syntax:

DngA_Qc = DangerAreaRico(RiscData)

Indata:

RiscData: Struktur med parametrar som beskriver riskområdet, se bilaga 1.

Utdata:

DngA_Qc: Matris av storlek *n*x2 där *n* blir godtyckligt stort beroende på indata. Matrisen innehåller koordinaterna som avgränsar riskområdet för eventuella ricocheter. Första kolonen representerar *x* koordinater och andra *y* koordinater.

OBS: All utdata är endast för positiva *x*. För att ta fram hela riskområdet får utdata speglas i linjen $x = 0$.

2.1.4.8 DangerAreaFrgm

Beräknar koordinaterna, *DngA_k*, som begränsar riskområdet som uppstår ifrån splitter enligt avsnitt 2.1.1.3 *Riskavstånd på grund utav splitter*. Funktionen använder sig av riskkoordinaterna beräknade i *DangerAreaRico*, avsnitt 2.1.4.7, samt *V*-linjens ekvation (se T1.e9).

Funktionen beräknar först koefficienterna, lutningen och konstanten, i räta linjens ekvation (se T1.e1) till linjerna som förbinder koordinaterna i *DngA_Qc* (*OBS:* endast positiva *x*).

2.1.4.8.e1

$$DngA\_Qc = \begin{bmatrix} x_1 & y_1 \\ \cdot & \cdot \\ \cdot & \cdot \\ \cdot & \cdot \\ x_n & y_n \end{bmatrix}$$

Detta görs med MATLAB-funktionen *polyfit* [8]. Utdata från *polyfit* är en matris *P* med de sökta koefficienterna enligt:

2.1.4.8.e2

$$P = \begin{bmatrix} k_1 & c_1 \\ \cdot & \cdot \\ \cdot & \cdot \\ \cdot & \cdot \\ k_{n-1} & c_{n-1} \end{bmatrix}$$

Matrisen *P* utökas sedan med koefficienterna för *V*-linjen (se T1.e9):

2.1.4.8.e3

$$P = \begin{bmatrix} k_v & 0 \\ k_1 & c_1 \\ \cdot & \cdot \\ \cdot & \cdot \\ \cdot & \cdot \\ k_{n-1} & c_{n-1} \end{bmatrix}$$

Därefter beräknas de nya linjerna som begränsar riskområdet ifrån splitter i enlighet med teorin i avsnitt 2.1.1.3. De nya linjerna har samma lutning som de i matrisen *P* men en ny konstant enligt ekvation T1.e21. Den nya matrisen *P´* med de nya konstanterna c_i´ förtydligas nedan:

2.1.4.8.e4

$$P' = \begin{bmatrix} k_v & c'_0 \\ \cdot & \cdot \\ \cdot & \cdot \\ \cdot & \cdot \\ k_{n-1} & c'_{n-1} \end{bmatrix}$$

Ett undantag till denna *P´* matris är om *cd*-värdet är angivet (se avsnitt 2.1.1.3, specialfall 3) och detta värde är större än riskavståndet ifrån splitter, *k*. Då beräknas c_0´ enligt:

2.1.4.8.e5

$$c'_0 = -\frac{sign(k_v) \cdot cd}{\cos(\arctan(k_v))}$$

Skärningspunkterna mellan linjerna i *P´* beräknas med ekvation T1.e22. Det är bland annat dessa skärningspunkter, [x_{si} y_{si}], i=1…n, som tillsist bygger upp matrisen *DngA_k*.

Eftersom att *DngA_k* endast innehåller koordinater som begränsar riskområdet från splitter för positiva *x*-värden ges den sista koordinaten, k_{end}, av [*0 h+k*].

Koordinaterna för den övre bågen i figur 1.8 (positiva x) beräknas med hjälp av T1.e4 enligt:

2.1.4.8.e5

$$\bar{x}_c = (h+k) \cdot \cos(\varphi)$$
$$\bar{y}_c = (h+k) \cdot \sin(\varphi)$$
$$\frac{\pi}{2} - (\alpha + V) < \varphi < \frac{\pi}{2}$$

Dessa koordinater placeras i *DngA_k* mellan k_{end} och skärningspunkterna [x_{si} y_{si}].

Till sist undersöks området kring origo som kan bildas enligt en del olika fall, se specialfall 1 till 3 i figur 1.9 avsnitt 2.1.1.3.

Är *f+k* lika med *Amin* och *k1* lika med noll börjar *DngA_k* (postiva *x*) i koordinaten k_{start} = [0 0]. Riskområdet fortsätter sedan parallellt med *f*-linjen till positionen $\left[(A\min - f) \cdot \tan(V+\alpha), 0\right]$ enligt figur 1.10 avsnitt 2.1.1.3. Därpå förlängs riskområdet i cirkelbågen enligt ekvation T1.e24:

2.1.4.8.e6

$$\bar{x}_{c2} = (A\min - f) \cdot \tan(V+\alpha) + k \cdot \sin(\beta)$$
$$\bar{y}_{c2} = k \cdot (1 - \cos(\beta))$$

Där β stegas från 0 till dess att residualen r_1 nedan blir noll.

2.1.4.8.e7

$$r_1 = \{(A\min - f) - k\cdot\cos(\beta)\} - \{P'(2,1)\cdot[(A\min - f)\cdot\tan(\alpha + V) + k\cdot\sin(\beta)] + P'(2,2)\}$$

Denna residual motsvarar skärningen mellan cirkelbågen nedtill i figur 1.10, samt begränsningslinjen för splitter som går parallellt med *Q*-linjen.

Är *k1* istället skilt från noll blir k_{start} = [*0 –k1*] och en ny cirkelbåge beräknas enligt ekvation T1.e25:

2.1.4.8.e8

$$\bar{x}_{c3} = k1\cdot\sin(\beta)$$
$$\bar{y}_{c3} = -k1\cdot\cos(\beta)$$

Denna gång stegas β från 0 till dess att en av residualen, r_1 eller r_2, i ekvation T1.e26 blir noll.

Stämmer inte någon av dessa specialfall in på splitterområdet börjar *DngA_k* i origo och fortsätter sedan i *N*-linjens riktning, riskområdet framför mynning, enligt ekvation T1.e23. Denna linje avslutas i punkten Nk_{end} = [Nk_{endX} Nk_{endY}], skärningen med *k*-linjen som går parallellt med *V*-linjen enligt figur 1.8.

De beräknade koordinaterna assembleras in i *DngA_k* och resulterar i:

2.1.4.8.e9, *Amin = f + k*

$$DngA\_k = \begin{bmatrix} 0 & 0 \\ (A\min - f)\cdot\tan(V+\alpha) & 0 \\ \bar{x}_{c2} & \bar{y}_{c2} \\ x_{s1} & y_{s1} \\ \cdot & \cdot \\ \cdot & \cdot \\ \cdot & \cdot \\ x_{sn} & y_{sn} \\ \bar{x}_c & \bar{y}_c \\ 0 & h+k \end{bmatrix}$$

2.1.4.8.e10, Ansatt *k1*-värde

$$DngA\_k = \begin{bmatrix} 0 & -k1 \\ \bar{x}_{c3} & \bar{y}_{c3} \\ x_{s1} & y_{s1} \\ \cdot & \cdot \\ \cdot & \cdot \\ \cdot & \cdot \\ x_{sn} & y_{sn} \\ \bar{x}_c & \bar{y}_c \\ 0 & h+k \end{bmatrix}$$

2.1.4.8.e11, Inverkan av *N*-linje

$$DngA\_k = \begin{bmatrix} 0 & 0 \\ Nk_{endX} & Nk_{endY} \\ x_{s1} & y_{s1} \\ \cdot & \cdot \\ \cdot & \cdot \\ \cdot & \cdot \\ x_{sn} & y_{sn} \\ \bar{x}_c & \bar{y}_c \\ 0 & h+k \end{bmatrix}$$

Syntax:

[DngA_k] = DangerAreaFrgm(RiscData)

Indata:

RiscData: Struktur med parametrar som beskriver riskområdet, se bilaga 1.

Utdata:

DngA_k: Matris av storlek *n*x2 där *n* blir godtyckligt stort beroende på indata. Matrisen innehåller koordinaterna som avgränsar riskområdet med hänsyns till eventuella splitter. Första kolonen representerar *x*-koordinater och andra *y*-koordinater.

OBS: All utdata är endast för positiva *x*. För att ta fram hela riskområdet får utdata speglas i linjen $x = 0$.

2.1.4.9 DangerAreaNu

Beräknar koordinaterna för riskområdet bakom vapnet som uppstår i samband med skottlossning enligt teorin i avsnitt 2.1.1.4. Cirkelbågen bakom vapnet ges av ekvation T1.e27 enligt:

2.1.4.9.e1

$$\bar{x}_c = u \cdot \sin(\beta)$$
$$\bar{y}_c = -u \cdot \cos(\beta)$$
$$-(N_u + \alpha) < \beta < N_u + \alpha$$

Matrisen *DngA_Nu* begränsar hela riskområdet bakom vapnet och ges av:

2.1.4.9.e2

$$DngA\_Nu = \begin{bmatrix} 0 & 0 \\ \bar{x}_c & \bar{y}_c \\ 0 & 0 \end{bmatrix}$$

Syntax:

DngA_Nu = DangerAreaNu(RiscData)

Indata:

RiscData: Struktur med parametrar som beskriver riskområdet, se bilaga 1.

Utdata:

DngA_Nu: Matris av storlek *n*x2 där *n* blir godtyckligt stort beroende på indata. Matrisen innehåller koordinaterna som avgränsar riskområdet bakom vapnet. Första kolonen representerar *x* koordinater och andra *y* koordinater.

2.1.4.10 DangerAreaMuzzel

Beräknar koordinaterna för riskområdet framför mynningen till vapnet som uppstår i samband med skottlossning. Detta område täcks oftast in av övriga delar av riskområdet och används av SäkI endast vid framställning av delat riskområdet (*SäkI G 2008, sida 62, Mynningsavstånd* [1]). Funktionen finns tillgänglig här för att illustrera effekterna av bland annat mynningsflamman. Delat riskområdet uppstår inte vid angivet *r*-värde.

Området består av en cirkelbåge framför vapnet enligt:

2.1.4.10.e1

$$\bar{x}_c = r \cdot \sin(\beta)$$
$$\bar{y}_c = r \cdot \cos(\beta)$$
$$-(N+\alpha) < \beta < N+\alpha$$

Matrisen *DngA_rN* ges av:

2.1.4.10.e2

$$DngA\_rN = \begin{bmatrix} 0 & 0 \\ \bar{x}_c & \bar{y}_c \\ 0 & 0 \end{bmatrix}$$

Syntax:

DngA_rN = DangerAreaMuzzel(RiscData)

Indata:

RiscData: Struktur med parametrar som beskriver riskområdet, se bilaga 1.

Utdata:

DngA_rN: Matris av storlek *n*x2 där *n* blir godtyckligt stort beroende på indata. Matrisen innehåller koordinaterna som avgränsar riskområdet framför mynningen till vapnet. Första kolonen representerar *x*-koordinater och andra *y*-koordinater.

2.1.4.11 DangerCase

Beräknar parametrarna *c* och *l* utifrån det aktuella riskfallet.

Riskfall	c	l
A	0,2 (*Dmax* – *Amin*)	0,8 *Dmax* – 0,7 *Amax*
B	0,15 (*Dmax* – *Amin*)	0,6 *Dmax* – 0,5 *Amax*
C	0,08 (*Dmax* – *Amin*)	0,4 *Dmax* – 0,3 *Amax*

Riskfall definierade ifrån SäkI G 2008 [1]

Riskfall kan även väljas till *other* då *c* och *l* matas in manuellt via en GUI.

Syntax:

[c, l] = DangerCase(RiscData)

Indata:

RiscData: Struktur med parametrar som beskriver riskområden, se bilaga 1.

Utdata:

c: Riskavstånd i sida vid studs
l: Riskavstånd bortom målområdet

2.1.4.12 RiscExamples

RiscExamples bestämmer innehållet i strukturen *RiscData* utifrån färdiga exempel. Funktionen innehåller sex olika *RiscData* strukturer för beskrivning av nedanstående exempel:

1) Skjutning med 8,4 cm granatgevär 86 och 8,4 cm spårljusövningsprojektil

2) Skjutning med skarpt pskott 86 spårljusspränggranat 86 modifierat

3) Skjutning i skog ak 5

4) Skjutning mot skog ak 5, *treeline* = Amin

5) Skjutning i öppen terräng ak 5

6) Skjutning med 8,4cm grg m/86 icke modifierad slpsgr

Syntax:

[RiscData, cancel] = RiscExamples(choise)

Indata:

choise: Skalär med värde 1 till 6 enligt ovan listade exemplen

Utdata:

RiscData: Struktur med parametrar som beskriver riskområdet, se bilaga 1.
cancel: Boolean lika med *false*

2.1.4.13 mainSakI

mainSakI är huvudprogrammet för beräkning av SäkI:s riskområden. Programmet exekverar ett av följande alternativ:

1. *RiscExamples*
2. *editRiscData*
3. *RiscData = RiscData0*

Fall 1 och 2 beskrivs under respektive funktionsbeskrivning. Vid exekvering av fall 3 beräknas riskområde utifrån strukturen *RiscData0* som är indata till *mainSakI*. Detta alternativ underlättar små ändringar i *RiscData* mellan riskområdesberäkningarna utan att behöva mata in samtliga inparametrar. Ändringarna görs då via MATLAB:s kommandofönster.

mainSakI kallar funktionen *DangerCase* för beräkning av parametrarna *c* och *l* och beräknar vinkeln α, om denna inte är given, utifrån bredden av målområdet *M* enligt ekvation T1.e5. Programmet kallar funktionen *calcRiscArea* för beräkning av riskområdet och använder tillsist funktionen *plotRiscArea* för att rita upp riskområdet.

Syntax:

[RiscAreaCoord, RiscData, SakIfig] = mainSakI(choise, RiscData0, exampleNbr)

Indata:

choise:	Val av exekvering enligt ovan, skalär 1-3
RiscData0:	Struktur med parametrar som beskriver riskområdet, se bilaga 1.
exampleNbr:	Anger vilket exempel som ska beräknas om *choise* = 1, se avsnitt 2.1.4.12.

Utdata:

RiscAreaCoord:	Struktur med koordinatmatriser för de olika områdena av det totala riskområdet, se bilaga 1.
RiscData:	Struktur med parametrar som beskriver riskområdet, se bilaga 1.
SakIfig:	”Handle” till figuren som innehåller plotten av SäkI:s riskområde.

2.2 Monte Carlo-simulerade riskområden

I de Monte Carlo-simulerade riskområdena definieras två olika typer av risker som uppstår vid skjutning:

- Passagerisk
- Splitterrisk

Passagerisk
Passagerisken är ett sannolikhetsvärde mellan 0 och 1 som anger sannolikheten att en projektil passerar ett visst område.

Splitterrisk
Splitterrisken är, i enlighet med passagerisken, ett sannolikhetsvärde mellan 0 och 1 som anger risken för skada ifrån splitter inom ett visst område.

Dessa risker bygger tillsammans upp det totala riskområdet som blir en sannolikhetsfördelning av summan ifrån passagerisken och splitterrisken. Summan tvingas till ett maxvärde på 1, vilket motsvarar en 100% risk.

Monte Carlo innehållsförteckning

2.2.1 Teori

Monte Carlo-simuleringar bygger på stora talens lag som säger att upprepade simuleringar med samma väntevärde kommer resultera i att medelvärdet av resultatet från samtliga simuleringar är nära väntevärdet. Eller mer matematiskt uttryckt från boken *Sannolikhetsteori och statistikteori med tillämpningar* s.128-129 [3]:

Låt $X_1, X_2 \ldots$ vara oberoende likafördelade stokastiska variabler med väntevärde μ, sätt sedan

T2.e1

$$\overline{X}_n = \sum_{i=1}^{n} X_i / n$$

Då gäller för alla $\varepsilon > 0$ att

T2.e2

$$P(\mu - \varepsilon < \overline{X}_n < \mu + \varepsilon) \to 1 \text{ då n} \to \infty$$

De stokastiska variablerna som förekommer i MC-simuleringarna är listade i tabell

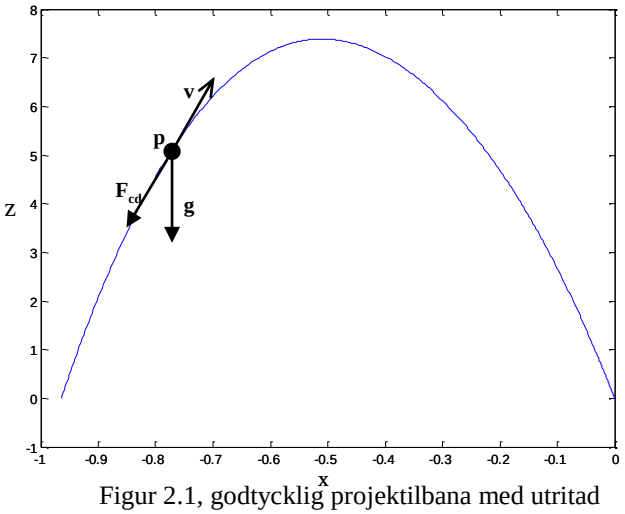
Tabell 2, Stokastiska variabler i Monte Carlo-simuleringarna

Stokastisk variabel	Fördelning	Väntevärde	Standardavvikelse
e0, elevationsvinkel från mynning	Normal, chi2, rektangel,	Elevationsvinkel från siktet	*Valfritt*
s0, vinkel i sida från mynning	Normal, rektangel	Vinkeln i sida ifrån siktet	*Valfritt*
s2, vinkel i sida före studs	Normal	vinkel i sida före studs beräknad ifrån v	*Valfritt*
e2, elevationsvinkel före studs	Abs(Normal)	elevationsvinkel före studs utifrån V_{uT} och V_{uN}	*Valfritt*
Bb1, banbrisad före armering	Binomial	*Valfritt*	*Valfritt*
Bb2, banbrisad efter armering	Binomial	*Valfritt*	*Valfritt*
Theta, vinkel för variation av studsytans normalriktning	Normal	0	Väljs godtyckligt efter riskfall
Phi, vinkel för variation av studsytans normalriktning	Likformigtfördelad mellan 0 och 2π	-	*Valfritt*

En simulering består av en beräknad projektilbana från avfyrning till eventuell brisering med sikte på ett förbestämt mål. Banans längd och utseende beror av de olika stokastiska variablerna listade ovan i tabell 2. Riskerna som uppkommer ifrån en simulering sparas och summeras med riskerna ifrån resterande *n* simuleringar i enlighet med T2.e1.

2.2.1.1 Projektilbanor

Figur 2.1 illustrerar utseendet av en projektilbana med utmarkerad projektil i punkt ***P***.
Projektilens hastighet och acceleration beskrivs enligt:



Figur 2.1, godtycklig projektilbana med utritad projektil P.

F_{cd} = *drag*
v = projektilens hastighet
g = gravitationens acceleration på projektilen

T2.e3

$$\bar{r} = \bar{r}(x, y, z, t)$$
$$\dot{\bar{r}} = \bar{v}(x, y, z, t)$$
$$\ddot{\bar{r}} = \bar{a}(x, y, z, t)$$

Där ***v*** är hastigheten och ***a*** accelerationen hos projektilen.

På grund utav luftmotståndet behöver flödesmekaniken kring projektilen beaktas. En kropp som utsätts för ett flöde i en godtycklig fluid utsätts för krafter och moment ifrån flödet. Om kroppen har en godtycklig form kommer dessa krafter och moment verka kring alla tre koordinatriktningar, $\boldsymbol{e_x}$, $\boldsymbol{e_y}$, $\boldsymbol{e_z}$. Om en axel väljs längs strömriktningen benämns kraften i denna riktning som *drag* och momentet kring denna axel *rolling moment*. Kraften vinkelrätt mot strömriktningen benämns *lift* och motsvarande moment *yaw*. Kroppen kan även utsättas för en tredje kraft i sidled och motsvarande *pitching moment* (*Fluid Mechanics* s.476-477 [4]). Kroppar med symmetriplan i förhållande till *lift-drag* planet påverkas däremot inte av *roll*, *yaw* eller krafter i sidled. Kroppar som har ytterligare symmetriplan påverkas tillslut endast av *drag* (*Fluid Mechanics* s.477 [4]). Projektiler faller in under de kroppar som har flera symmetriplan och utsätts därför endast av *drag* ifrån flödet parallellt med planen.

Genom att ansätta en projektilmassa *m*, och med hänsyns till friläggningen i figur 2.1 kan kraftekvationen för projektilen nu skrivas enligt nedan:

T2.e4

$$m \cdot \bar{a} = \bar{F}_{cd} + m \cdot \bar{g}$$
$$\bar{g} = -g \cdot \bar{e}_z$$
$$\bar{F}_{cd} = -D \cdot \frac{\bar{v}}{|\bar{v}|}$$

$\boldsymbol{F_{cd}}$ är luftens *drag* på projektilen och ***g*** är gravitationens acceleration. *D* (*drag*) är beloppet av $\boldsymbol{F_{cd}}$ och kan utryckas:

T2.e5

$$D = \frac{1}{2} \cdot \rho \cdot V^2 \cdot A \cdot c_D$$ (*Fluid Mechanics* s.478 [4])

D beror på variablerna:

ρ: densiteten i fluiden
V: projektilens fart
A: projektilens karakteristiska area (*Fluid Mechanics* s.478 [4])
c_D: projektilens *drag* koefficient (*Fluid Mechanics* s.476 [4])

Luften antas i fortsättning vara en ideal gas (*Fluid Mechanics* s. 601 [4]) (A.1). Detta resulterar i att trycket kan skrivas som $p = \rho RT$, där *R* är gaskonstanten och *T* är temperaturen.

Machtalet, *M*, ges av:

T2.e6

$$M = \frac{V}{M_1}$$

Där M_1 är ljudhastigheten som kan uttryckas:

T2.e7

$$M_1 = \sqrt{R \cdot k \cdot T}$$

För att undvika *ρ* i ekvation T2.e5 skrivs *D* nu istället om till en funktion av trycket, *p*, och machtalet, *M*, enligt:

T2.e8

$$D = \frac{k \cdot p \cdot M^2}{2} \cdot A \cdot c_D$$

Omskrivningen ger att *D* beror på variablerna:

k: det specifika värmeförhållandet
p: tryck
M: machtal
A: projektilens karakteristiska area
c_D: projektilens *drag* koefficient.

Omskrivning av T2.e4 med hjälp av T2.e3 ger följande differentialekvation för projektilens rörelse

T2.e9

$$\ddot{\bar{r}} = \frac{\bar{F}_{CD}}{m} + \bar{g} = -\frac{D}{m} \cdot \frac{\dot{\bar{r}}}{\left|\dot{\bar{r}}\right|} + \bar{g}$$

Vissa projektiler som ska studeras har även raketdrift. Detta resulterar i ytterliggare en acceleration i rörelseekvationen. Denna acceleration, ***R***, riktas i rörelseriktningen. Vid raketdrift minskar projektilens *drag*. Förklaring bakom detta är relativt komplicerad och kräver en hel del kunskap inom flödesmekanik och är inget som analyseras närmare i denna rapport. Enkelt sammanfattat minskas c_D på grund utav att gasutströmningen bakom projektilen ändrar tryckförhållanden i flödet.

Slutligen fås två rörelseekvationer:

T2.e10

$$\ddot{\bar{r}} = -\frac{D}{m} \cdot \frac{\dot{\bar{r}}}{\left|\dot{\bar{r}}\right|} + \bar{g}$$

$$\ddot{\bar{r}} = (\bar{R} - \frac{D_{reduc}}{m}) \cdot \frac{\dot{\bar{r}}}{\left|\dot{\bar{r}}\right|} + \bar{g}$$

Där D_{reduc} är det reducerade motståndet ifrån flödet vid raketdrift. Flödet som projektilen utsätts för är den som uppkommer på grund utav att projektilen förflyttar sig igenom luften.

2.2.1.2 Kollisionseffekter

Den största svårigheten vid simulering av riskområden är att ta fram en trovärdig modell för stöteffekter. En stöt kan resultera i:

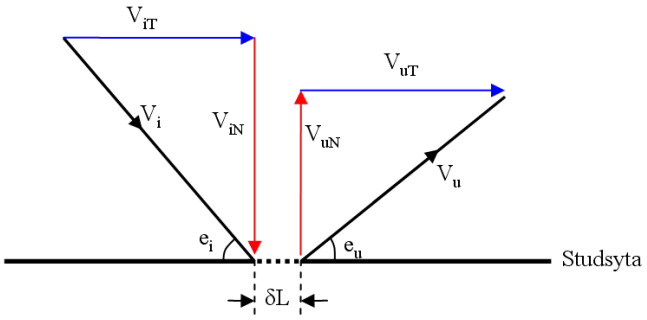
a) Riktningsändring
b) Rotationer
c) Brisad

a) Riktningsändring

En kropp med en given hastighet och massa har en bestämd rörelsemängd, ***p***, enligt:

T2.11

$$\bar{p} = m \cdot \bar{v}$$



Figur 2.2, godtyckligt studsfall

Rörelsemängden har en bestämd storlek och riktning. Lagen om rörelsemängdens bevarande, som en konsekvens av Newtons första och andra lag (*Mekanik Grundkurs* s. 165-167 [7]), säger att rörelsemängden för ett slutet system är lika före och efter en stöt om inga yttre krafter påverkar systemet. Ändringen av rörelsemängden för en godtycklig kropp är lika med impulsen, ***I***, och beräknas enligt:

T2.12

$\bar{I} = \int_{t_0}^{t_1} \bar{F} \cdot dt$, t_0 till t_1 är tiden kraften ***F*** får verka på kroppen

Vid beräkningar av stöteffekter vid kollisioner med till exempel markytor kan det vara svårt att ta hänsyn till samtliga delar av systemet och samtidigt ha någorlunda enkla och snabba beräkningar. Stöten förenklas därför till två delar. Den inkommande hastigheten uppdelas i en tangentiell hastighet och en normalriktad hastighet i förhållande till studsytan (se figur 2.2).

Till detta definieras följande hastigheter:

V_i: infallshastigheten före studs
V_{iT}: Tangentiella delen av ***V_i*** i förhållande till studsytan
V_{iN}: Vinkelräta delen av ***V_i*** i förhållande till studsytan
V_u: Utgångshastigheten efter studs
V_{uT}: Tangentiella delen av ***V_u*** i förhållande till studsytan
V_{uN}: Vinkelräta delen av ***V_u*** i förhållande till studsytan

Ifrån dessa hastigheter definieras följande parametrar:

T2.e13

$$\alpha = \frac{\left|\overline{V}_{uT}\right|}{\left|\overline{V}_{iT}\right|} = \frac{\left|\overline{V}_{u}\right| \cdot \cos(e_u)}{\left|\overline{V}_{i}\right| \cdot \cos(e_i)} \Rightarrow 0 \le \alpha \le 1$$

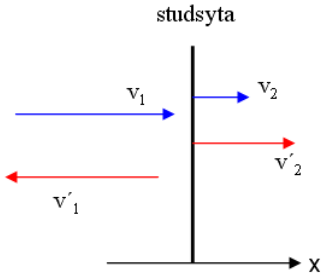
$$\beta = \frac{\left|\overline{V}_{uN}\right|}{\left|\overline{V}_{iN}\right|} = \frac{\left|\overline{V}_{u}\right| \cdot \sin(e_u)}{\left|\overline{V}_{i}\right| \cdot \sin(e_i)} \Rightarrow 0 \le \beta \le 1$$

α kan ses som en typ av friktionsparameter som beror på sträckan δL i figur 2.2. Ju längre δL desto längre tid har friktionen ifrån studsytan att verka på den tangentiella hastigheten vilket resulterar i ett mindre α-värde enligt T2.e13. Ändringen av den vinkelräta hastigheten i förhållande till studsytan analyseras som en studs mot en orörlig yta.

Studstalet, ε, definieras enligt (*Mekanik Grundkurs* s. 234, ekvation 9.10 [7]):

T2.e14

$$\varepsilon = -\frac{v_2' - v_1'}{v_2 - v_1}$$



Figur 2.3, stötförlopp

Där v_1 och v_2 är projektilfarten respektive studsytans fart före stöt och v_1' samt v_2' är de respektive farterna efter stöt (se figur 2.3). Studsytans fart före och efter stöt är lika med noll och studstalet reduceras till:

T2.e15

$$\varepsilon = -\frac{v_1'}{v_1}$$

Vilket är, med skillnad på tecken, hur parametern β definierades i ekvation T2.e13.

I simuleringarna är utfallshastigheten inte känd vilket gör att studstalet inte kan beräknas enligt T2.e15. Minskningen av hastigheten vid stöten i figur 2.3 kommer att bero på projektilens samt studsytans egenskaper. Projektilen kommer till exempel att deformeras vid tillräckligt stora infallshastigheter varav en del av den inkommande mekaniska energin går åt till arbetet som krävs för deformationen. Studsytan kan även vara mer eller mindre solid vilket dels påverkar deformationen av projektilen men även resulterar i att v_2' inte alls behöver vara noll för hela systemet. Hur β beror på dessa egenskaper kan uppskattas med hjälp av utomstående simuleringar för specifika islagsmaterial och projektiler (A.2).

Projektilen kan utöver att deformeras vid stöt brytas sönder till relativt stora delar som var och en fortsätter i en ny riktning efter kollisionen.

För att simuleringar av studseffekterna som beskrivits ovan ska kunna ge trovärdiga resultat behöver gränser för infallsvinklar och hastigheter där projektilen med stor sannolikhet bryts sönder eller briserar definieras utifrån utomstående simuleringar eller experiment.

b) Rotationer
Rotationer efter kollision kommer inte studeras i detta examensarbete. Antagandet blir istället att största riskerna uppstår vid rena riktningsändringar (A.3). Detta resulterar i att projektilen bör nå en längre distans efter studs jämfört med verkligheten eftersom den då inte förlorar energi till rotation.

c) Brisad
Vid en kollision påverkas projektilen av en impuls. Storleken och riktningen av kraften i impulsen avgör om projektilen briserar eller inte. Impuls beräknas som integralen av kraften med avseende på tiden enligt ekvation T2.12. Impulsen kan även beräknas som skillnaden i rörelsemängd för ett objekt:

T2.e16

$\bar{I} = m \cdot \bar{v}_1 - m \cdot \bar{v}_0$, $\boldsymbol{v_0}$ och $\boldsymbol{v_1}$ är hastigheten före respektive efter stöt

Vid en kollision antas att projektilen är orienterad i hastighetens riktning, nosen pekar alltså åt samma håll som $\boldsymbol{V_i}$ (A.4), se figur 2.2. Den avgörande parametern för impulsens riktning i förhållande till projektilens orientering blir alltså infallsvinkeln e_i mot kollisionsytan. Den andra avgörande faktorn är under vilken tid, t_0 till t_1, impulsen verkar. Denna beror på egenskaper hos projektilen samt studsytan.

För att en brisad ska äga rum behöver kollisionen ske med en infallsvinkel e_i vilken överstiger en förbestämd vinkel e_{imax} som beror på studsytan och projektilens ”hårdhet”.

2.2.1.3 Skjutområdet

I Monte Carlo-simuleringarna avfyras projektiler över ett område som till storlek bestäms utifrån typen av projektil som ska analyseras. Topografins egenskaper, så som markens beskaffenheter, bestäms fritt från fall till fall.

Skjutområdets topografi består utav en mängd mindre kvadratiska topografielement enligt figur 2.4 samt 2.5. Antal element bestäms utav skjutområdets längd i *x*-led, *l*, bredd i *y*-led, *b*, samt elementens sidlängd, *sz*. Antalet element i topografin är:

T2.e17

$$nelm = \frac{l \cdot b}{sz^2}$$

Dessa element är numrerade 1 till *nelm* utgående från det övre vänstra elementet enligt figur 2.6. För att beskriva markens beskaffenheter tilldelas varje element en parameter *rhog* som anger markens ”hårdhet”.

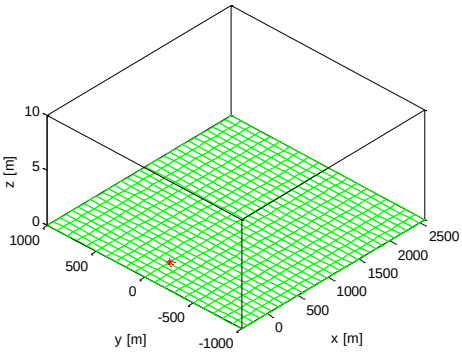
I topografin skapas även noder. Dessa utgör skärningspunkterna för tänkta longitudinella samt latitudinella linjer som bildar elementens sidor, se figur 2.7. Antalet noder i given topografi är:

T2.e18

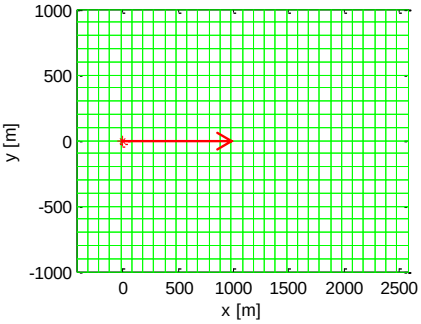
$$non = \left(\frac{l}{sz} + 1\right) \cdot \left(\frac{b}{sz} + 1\right)$$

Noderna är numrerade på liknande sätt som elementnumreringen och börjar i övre vänstra hörnet enligt figur 2.7. Dessa noder utgör lagringspunkter för riskerna som uppstår i simuleringarna.

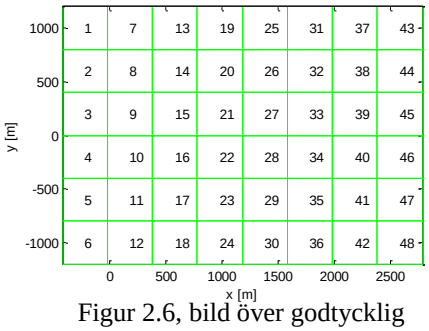
Elementen och noderna är sammankopplade och varje element har fyra tillhörande noder (hörnen). Noderna fastställs till ett element med en bestämd orientering. Till element 1 i figur 2.6 tillhör noderna 1, 2, 9 och 8, enligt figur 2.7. Element 2 får på motsvarande sätt



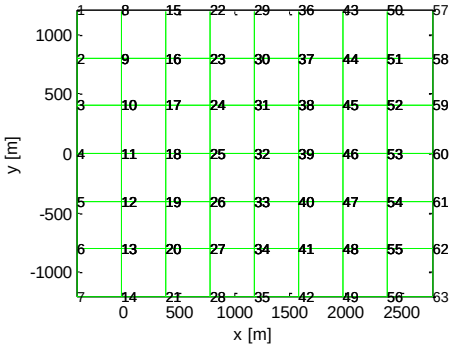
Figur 2.4, bild över godtycklig topografi snett uppifrån, röd stjärna markerar origo.



Figur 2.5, bild över godtycklig topografi rakt uppifrån, röd stjärna markerar origo, röd pil markerar skjutriktningen.



Figur 2.6, bild över godtycklig topografi med utmarkerad elementnumrering



Figur 2.7, bild över godtycklig topografi med utmarkerad nodnumrering

noderna 2, 3, 10 och 9. Elementens nodorientering går alltså moturs med start i elementens övre vänstra hörn enligt figur 2.7. Denna orientering är viktig att hålla reda på när risker ska ansättas och analyseras och beskrivs närmare i avsnitt 2.2.5.4: *createTopografi4*.

I topografin placeras målet på linjen $y = 0$. Målet har en bestämd utbredning i *y*- och *z*-led. Detta bildar en imaginär vägg med bestämd höjd och bredd men utan djup.

2.2.1.4 Passagerisk

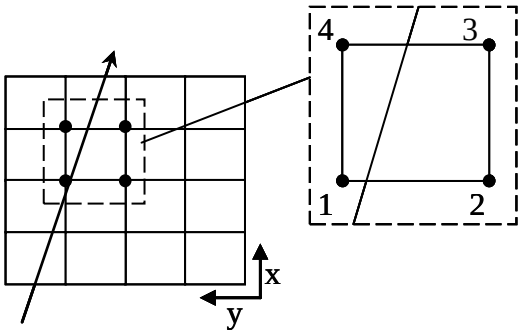
När en projektil färdas genom rummet skapas en projektilbana. Passagerisken beräknas utifrån banan, som projektionen av banan ner på marken. I Monte Carlo-simuleringarna byggs denna risk upp och bildar ett riskområde för passage som illustrerar var projektilerna har färdats och även med vilken frekvens de har färdats över området.

Passagen av en projektil lagras i topografins noder. Risken i en bestämd nod beräknas med interpolation och beror på avståndet till den nedprojicerade projektilbanan samt antalet noder i topografin.

Är projektilen på tillräcklig höjd över marken kan risken vid marknivå antas försvinna. Denna höjd, h_{min}, bestämmer den lägsta höjd över vilken en projektil inte utgör risk vid marknivå. Höjden h_{min} bör inte vara mindre än *k*-värdet för projektilen. *k*-värdet anger det största avstånd bortanför vilken splitterriken från brisad är noll och beskrivs närmare i avsnitt 2.2.1.6: *Splitterrisk.*

Efter varje simulerad bana sparas projektilens färd i en riskmatris. Riskmatrisen är av storleken [*non*,1], där *non* är lika med antalet noder i topografin enligt ekvation T2.e18. Vid passage över ett visst element ska noderna tilldelas riskvärden för passage. Detta riskvärde är 1 för passage rakt över en nod, samt 0 då projektilen ej passerar förbi elementen som tillhör noden (en nod tillhör fyra olika element med undantag för noderna längs topografins gränser).

Figur 2.8 illustrerar passage över ett element med utmarkerade noder. Riskvärden interpoleras till noderna med Lagrange interpolation.



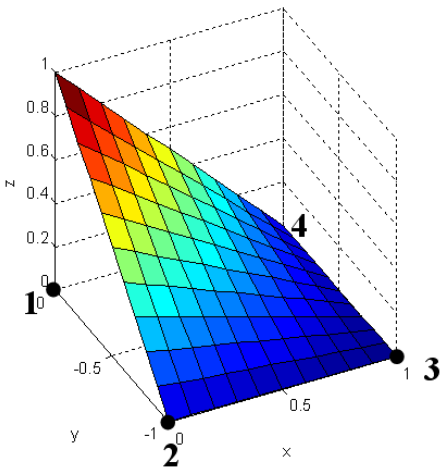
Figur 2.8, exempel på passage över ett element i topografin

Lagrange interpolation

I varje nod skapas en ”formfunktion”, dessa formfunktioner har egenskaperna att de har värdet 1 i noden de tillhör samt 0 i resterande noder. De har även egenskapen att de bara existerar i elementet som undersöks. Studerar vi nod 1 i figur 2.8 ger Lagrange interpolationsformel i *x*-led *(”Introduction to F.E.M” s. 127* [5]), med viss modifikation:

T2.e19

$$l_1(x) = -\frac{x - x_4}{x_1 - x_4}$$



Figur 2.9, formfunktion N_1 för element med sz=1

Denna funktion uppfyller villkoren: $l_1(x_1)=1$, samt $l_1(x_4)=0$. Motsvarande formfunktion i *y*-led blir:

T2.e20

$$l_1(y)=\frac{y-y_2}{y_1-y_2}$$

Som uppfyller $l_1(y_1)=1$, samt $l_1(y_2)=0$.

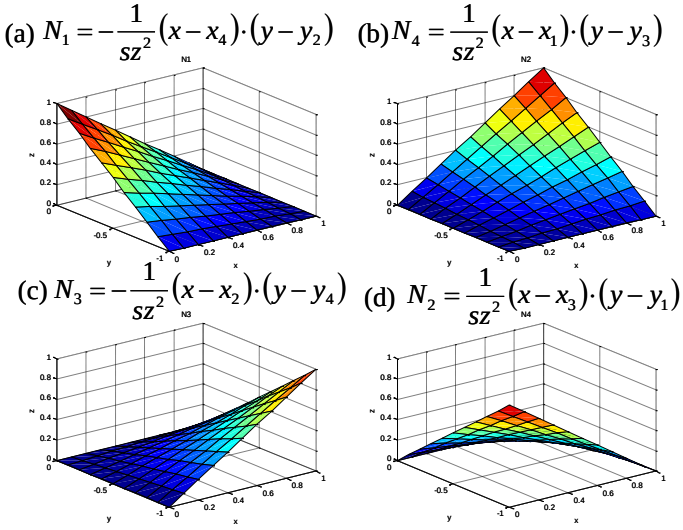
Genom att multiplicera ihop dessa funktioner fås formfunktionen för nod 1 enligt:

T2.e21

$$N_1=l_1(y)\cdot l_1(x)=-\frac{x-x_4}{x_1-x_4}\frac{y-y_2}{y_1-y_2}=-\frac{1}{sz^2}(x-x_4)\cdot(y-y_2)$$

N_1 uppfyller nu samtliga villkor. I figur 2.9 är N_1 utritad för ett element med *sz = 1* samt nodkoordinaterna: [*0 0*], [*0 -1*], [*1 1*] och [*1 0*] i *xy*-planet.

Övriga formfunktioner skapas på motsvarande sätt. I figur 2.10 visas samtliga formfunktioner över ett element. Summan av formfunktionerna över elementet resulterar i planet *z = 1*. Med dessa funktioner kan passagerisken interpoleras till noderna genom att föra in projektilens bana i *xy*-planet till formfunktionerna.



Figur 2.10, formfunktioner för samtliga noder hos ett element med sz = 1, a) nod 1, b) nod 4, c) nod 3, d) nod 2

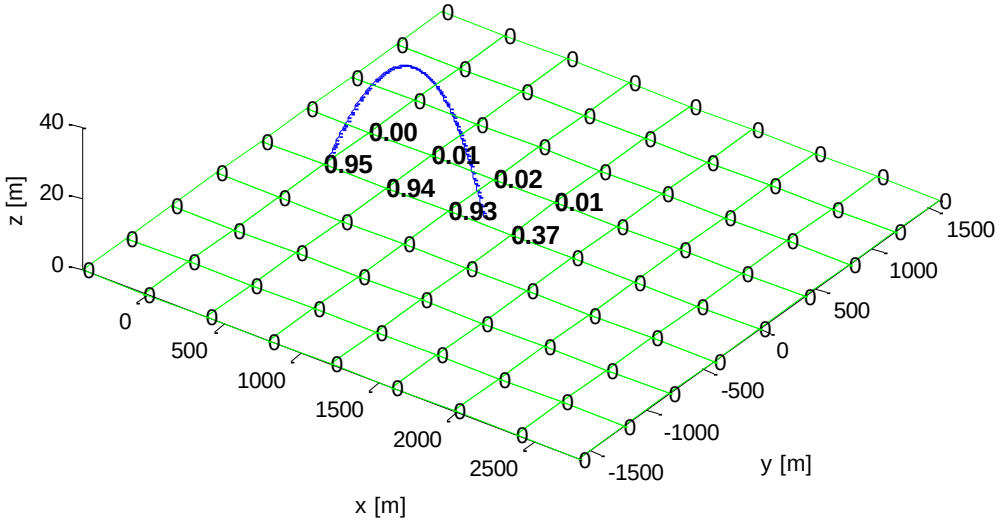
Denna metod kräver att projektilbanan är diskretiserad till ett antal koordinater som kan föras in i formfunktionerna.

Risken som lagras i en nod från en given projektilbanan är det största riskvärdet ifrån interpolationen av de diskreta punkterna. Passagerisken för en viss nod beror alltså endast på det kortaste avståndet till projektilbanan.

För att interpolera fram passagerisken enligt ovan bör koordinaterna i ***p*** vara jämnt fördelade i *xy*-planet. Är koordinaterna inte jämnt fördelade, eller för glest fördelade kan de ge en sned bild av passagerisken. Är punkterna istället väldigt tätt fördelade i förhållande till elementstorleken medför detta att interpolationsberäkningarna för passagerisken tar onödigt lång tid då de enda egentligt intressanta koordinaterna är de som ligger närmast noderna. Dessa problem löses genom att beräkna jämnt fördelade koordinater i den ursprungliga projektilbanan genom att approximera förflyttningen i

xy-planet till en rät linje mellan utgångsläge och nedslagspunkt. Förekommer studs effekter i projektilbanan approximeras flera linjer, en för varje ”delbana”. Jämnt fördelade punkter på linjen tas sedan ut och för att få en bra riskfördelning bör punkternas inbördes avstånd göras *tillräckligt* litet (<*sz/2*).

I figur 2.11 nedan visas ett exempel på hur riskområdet för passage fördelas till de topografiska noderna.



Figur 2.11, exempel på passagerisk för given projektilbana, blå parabel.

En verkan av interpolationen är att projektilbanan ”breddas” från att vara en linje till ett område. Denna bredd i sida, eller ”säkerhetsavstånd”, λ_0, ligger inom området:

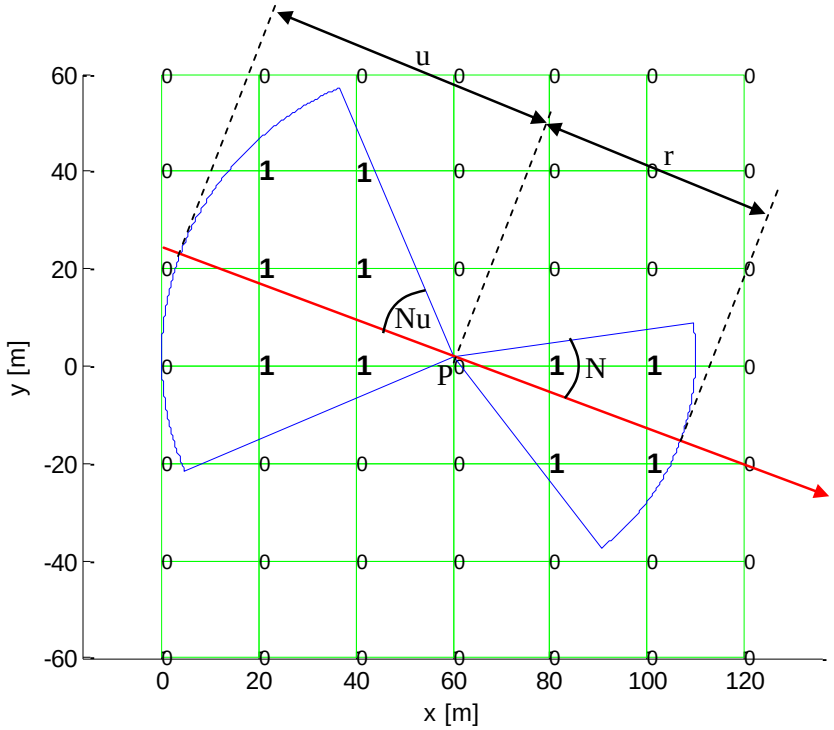
T2.e22

$$sz < \lambda_0 < \sqrt{2} \cdot sz$$

Denna uppkommer eftersom att det är ”globala” noder som används, varje nod tillhör mer än ett element. I slutresultatet syns det inte vilket element som har tilldelat en viss nod en viss risk. Storleken på λ_0 beror på var och i vilken vinkel projektilbanan skär elementgränserna. Risken från en bestämd projektilbana vid λ_0 är noll och ökar enligt formfunktionerna mot banans ursprungliga linje. Detta illustreras tydligt i figur 2.11.

2.2.1.5 Riskområde från mynning samt bakom vapnet

Riskområdena som uppstår framför mynning samt bakom vapnet skapas i enlighet med SäkI:s riskområden, (*SäkI G 2008* s. 61-62 samt 63-63 [1]). Två cirkelsektorer skapas utifrån avfyrningspositionen. Framför mynningen används vinkel *N* samt det radiella avståndet *r*. Bakom vapnet motsvarande vinkel *Nu* och radiellt avstånd *u*. I figur 2.12 visas de två områdena. Observera skillnaden i definitionen av vinklarna i förhållande till *SäkI G 2008* [1]. Noderna innanför dessa områden tilldelas riskvärdet 1.



Figur 2.12, riskområde framför mynning samt bakom vapnet, skjutriktning i den röda pilens riktning. I detta exempel ligger avfyrningspositionen, **P**, strax ovanför origo enligt figur.

2.2.1.6 Splitterrisk

Vid brisad beräknas ett riskområde för splitter. Området som bildas beror på projektilens orientering i rymden vid briseringsögonblicket, projektilens uppbyggnad samt terrängen. För att täcka in samtliga riskfaktorer vid brisering skapas cirkulära områden med centrum i briseringspunkten.

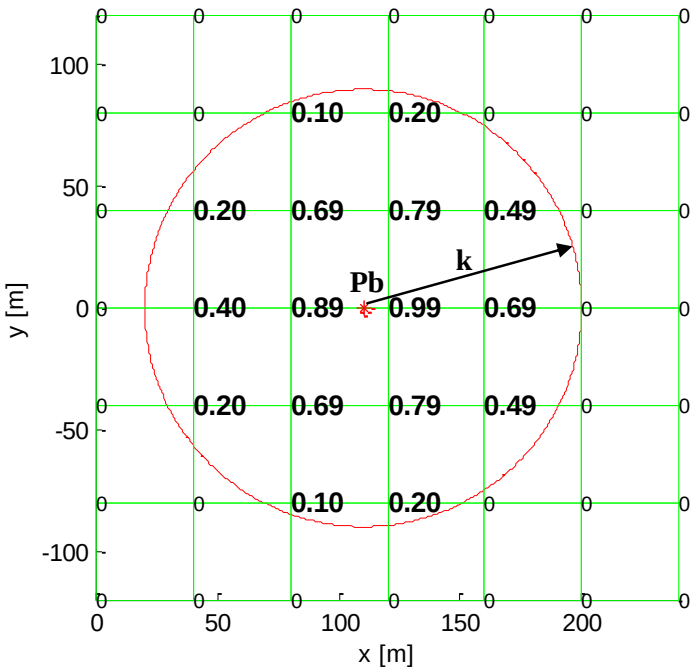
Riskvärden appliceras till noderna i topografin (se avsnitt 2.2.1.3: *Skjutområdet*) inom splitterområdet som funktion av avståndet till briseringspunkten. Riskvärdet vid briseringspunkten är 1 och vid $r = k$ är riskvärdet 0. r är det radiella avståndet ifrån briseringspunkten och k är det största avstånd bortanför vilken splitter inte utgör risk. k-värdet är förbestämt för varje typ av projektil och motsvarar k-värdet definierat i SäkI (*SäkI G, 2008*, s. 62 [1]).

Risken avtar radiellt utåt ifrån briseringspunkten. Minskning av risken är kvadratiskt med hänsyn till att arean splittret ska verka över ökar kvadratiskt radiellt ifrån briseringspunkten. Risken för en given nod inom splitterområdet blir:

T2.e23

$$Risk = 1 - \frac{a^2}{k^2}$$

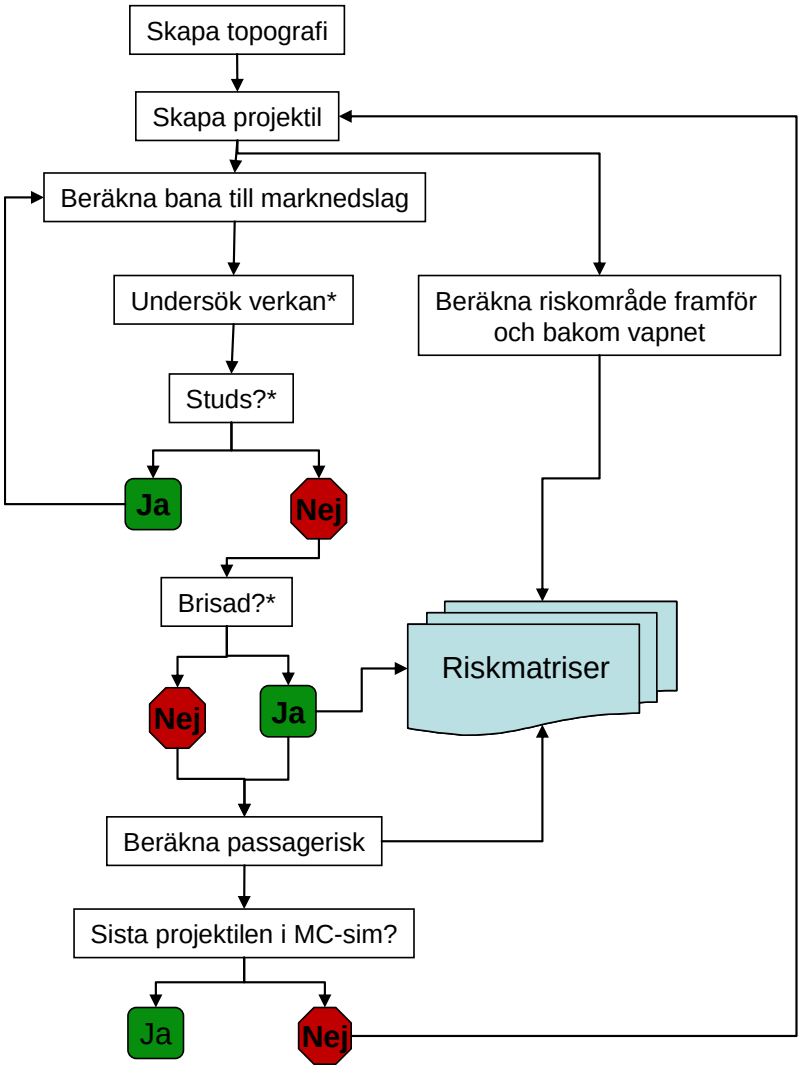
där a är det radiella avståndet till briseringspunkten. Figur 2.13 illustrerar riskfördelning vid brisad.



Figur 2.13, riskområde vid brisad i position **Pb** med maximalt riskavstånd för splitter lika med k

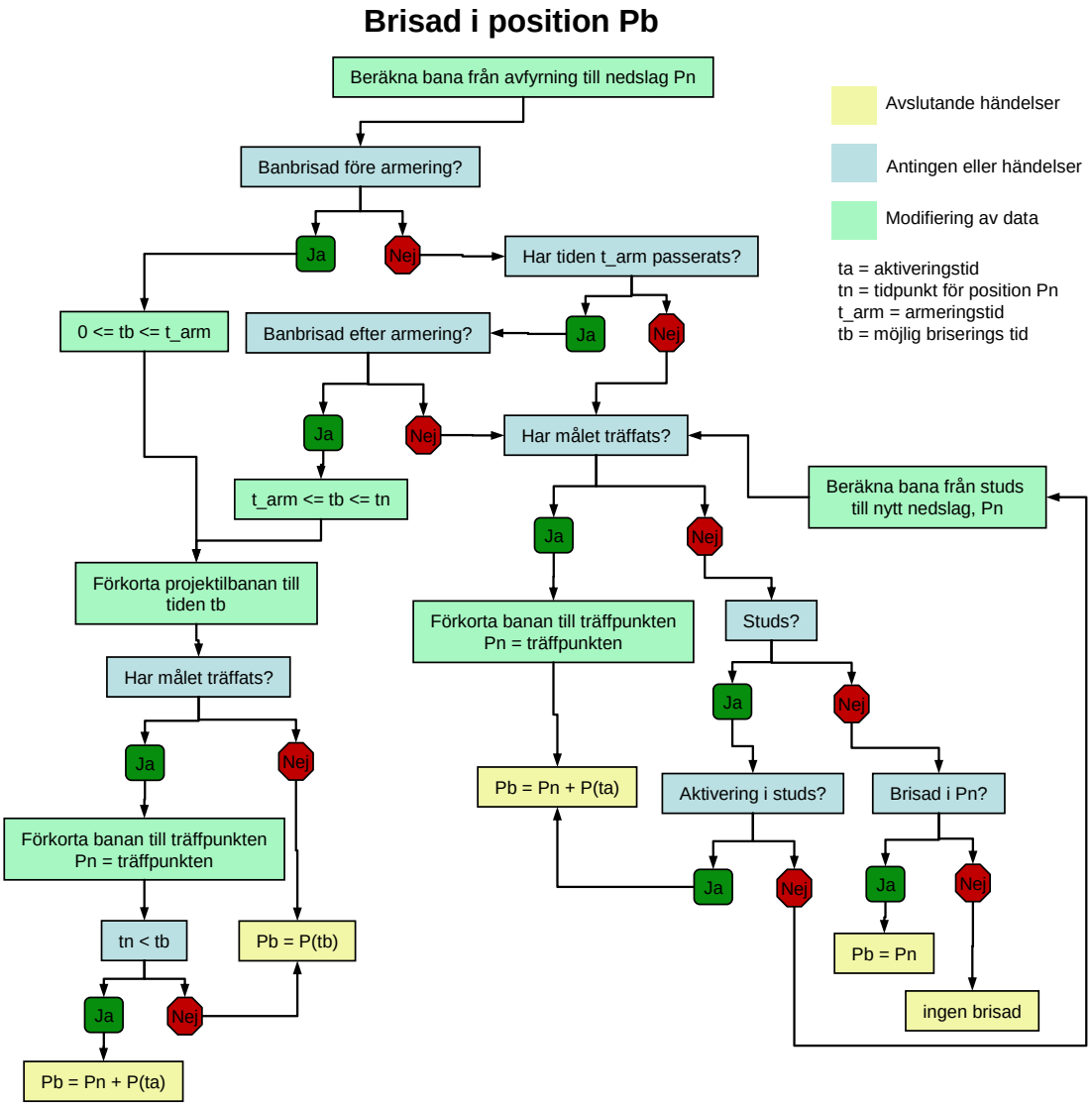
Vid brisering över marknivå beräknas samma cirkulära område som ovan. Detta område projiceras sedan ned i *xy*-planet.

2.2.2 Monte Carlo-Flödesschema



Flödesschemat ovan ger en överskådlig bild av stora delar av MC-simuleringen. De delar som är markerade med (*) kräver mer förklaring och beskrivs i ett separat flödesschema längre fram i detta avsnitt.

En projektilbana sträcker sig ifrån en given startposition tills dess att projektilen slår i marken. Denna projektil kan sedan fortsätta i en ny bana enligt flödesschemat ovan (studseffekt, se avsnitt 2.2.1.2). En fullständig projektilbanan avslutas genom att projektilen antingen briserar eller då projektilens fart är noll. Under denna färd utsätts projektilen för olika risker och händelser enligt flödesschemat *Brisad i position Pb* nedan.



Risker och händelser som kan inträffa under projektilfärden är:

1) Brisering på grund utav konstruktionsfel
    a. Före armering ⇨ *tb < t_arm*
    b. Efter armering ⇨ *tb > t_arm*
2) Träff av mål ⇨ *projektilen aktiveras*
3) Studs
    a. Aktivering ⇨ *Pb = Pn + P(ta)*
    b. Ej aktivering ⇨ *ny bana*
4) Tidsbestämd detonation ⇨ *Pb = P(dtimer)*
5) ”Blindgångare”

tb = briseringstid
t_arm= armeringstid
ta = aktiveringstid
Pb = brisadpunkt
Pn = nedslagspunkt
dtimer = detonationstimer

1) Brisering på grund utav konstruktionsfel

Det finns risk för brisering på grund utav konstruktionsfel i projektilbanan. Denna är binomialfördelad över två delar av banan. Den första uppkommer före armeringstiden *t_arm*, och den andra efter armeringstiden. Var i banan denna brisering sedan sker är rektangelfördelat i tiden.

2) Träff av mål
Har projektilbanan skurit ”väggen” som skapas i topografin (se avsnitt 2.2.1.3: *Skjutområdet*) förkortas projektilbanan till skärningspunkten och projektilen aktiveras och briserar efter given aktiveringstid.

3) Studs
Studseffekten beskriven i teorin, avsnitt 2.2.1.2, undersöks och de utgående vinklarna och hastigheterna avgör vad som hänt i studsen. Här finns ett par olika alternativ. Den första, och kanske mest sannolika händelsen är att projektilen ”aktiveras” vid nedslag. En aktiverad projektil briserar efter bestämd aktiveringstid. Denna händelse inträffar om infallsvinkel mot föremål/mark överstiger förbestämd maximal studsvinkel. Är infallsvinkeln inte tillräckligt stor för att projektilen ska aktiveras beräknas en ny bana med utgångshastighet ifrån studsberäkningen.

4) Tidsbestämddenotation
Har projektilen denna egenskap briserar den då simuleringen når tiden *dtimer*.

5) ”Blindgångare”
Projektilen saknar tidsbestämddetonation och dess fart har upphört utan att någon av briseringskraven har uppfyllts.

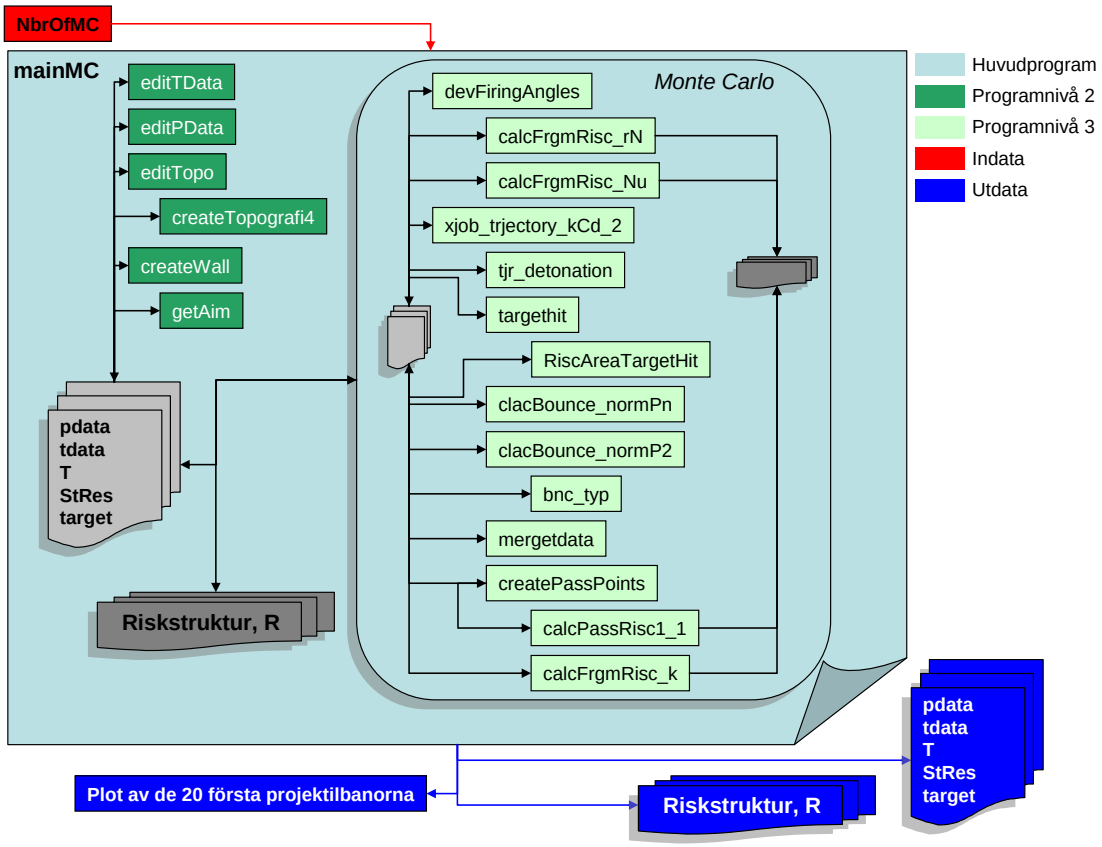
Hur dessa händelser infaller i simuleringen framgår i flödesschemat *Brisad i position Pb* ovan.

Utfallet ifrån en simulering enligt de två flödesscheman ovan är riskmatriser med riskvärden för passage (avsnitt 2.2.1.4), riskvärden ifrån riskområdet framför samt bakom vapnet (avsnitt 2.2.1.5) samt riskvärden ifrån splitterutbredningen vid eventuell brisad (avsnitt 2.2.1.6). Riskerna sparas och summeras sedan med riskerna från resterande simuleringar i enlighet med teorin i avsnitt 2.2.1.

2.2.3 Monte Carlo-simulering i MATLAB

Nedan visas ett schema över hur programmet för Monte Carlo-simulerade riskområden arbetar.

Programstruktur för MC riskområdesberäkningar



För att beskriva projektilen, projektilbanan, topografin och målet skapas fyra strukturer med parametrar som anger respektive egenskaper:

Strukturer

Benämning	Beskriver
pdata	Projektilen
tdata	Projektilbanan
T	Topografin
target	Målet

Två ytterligare strukturer skapas för att beskriva händelser och risker som uppkommer under simuleringen.

Strukturer

Benämning	Innehåll
StRes	Statistiska resultat
R	Risker

Samtliga strukturer återfinns i bilaga 1: *Struktursamling*.

Funktionerna under de olika programnivåerna i schemat *Programstruktur för MC riskområdesberäkning* ovan finns beskrivna under avsnitt 2.2.4 samt 2.2.5.

2.2.4 Monte Carlo-funktioner

Följande funktioner används vid beräkning av Monte Carlo-simulerade riskområden i MATLAB.

Tabell 3, funktioner för beräkning av Monte Carlo-simulerade riskområden i MATLAB

Benämning	Beskrivning
bnc_type	Kontrollerar vilken typ av studs som skett
calcAlphaBeta	Beräknar studsparametrarna α och β
calcFrgmRisc_k	Beräknar briseringsområde
calcFrgmRisc_Nu	Beräknar riskområde bakom vapnet
calcFrgmRisc_rN	Beräknar riskområdet framför mynning
calcPassRisc1_1	Beräknar passagerisk
clacBounce_normP2	Beräknar studs med normalfördelade elevationsvinklar och sidvinklar
clacBounce_normPn	Beräknar studs genom att deviera studsytans normalriktning
createPassPoints	Skapar passeringspunkter till passagerisken
createTopografi4	Skapar topografistrukturer med kvadratiska fyrnodselement
createWall	Skapar målstrukturen
devFiringAngles	Beräknar avvikelsen i skjutriktningen vid avfyrning
editPData	Funktion för att ändra projektildata
editTData	Funktion för att ändra projektilbandata
editTopo	Funktion för att ändra topografiparametrar
formfunc	Beräknar Lagrange interpolationen för passagerisken
getAim	Beräknar vilket sikte som krävs för att träffa givet mål
getDtpos	Beräknar briserings position utifrån en given tidpunkt
getElmNbr	Ger element nummer för viss position
insertRisc	Assemblerar in lokalelement risk till den globala riskmatrisen
mainMC	Huvudprogram
mergetdata	Summerar ihop data från två projektilbanor
RiscAreaTargetHit	Beräknar riskområdet som uppstår vid träff av målet som skapats med *createWall*
targethit	Kontrollerar om målet har träffats
tjr_detonation	Beräknar banbrisader som uppstår pga konstruktionsfel
xjob_trjectory_kCd_2	Beräknar projektilbanan

2.2.5 Funktionsbeskrivningar

Nedan beskrivs de funktioner som används vid beräkning av Monte Carlo-simulerade riskområden i MATLAB.

2.2.5.1 mainMC

mainMC är huvudprogrammet för Monte Carlo-simuleringarna. Härifrån anropas övriga funktioner för beräkningar i MC-simuleringarna. I *mainMC* skapas även två strukturer för redovisning av resultat, riskstrukturen *R* samt strukturen *StRes*.

Riskstruktur *R*

Riskstrukturen *R* innehåller fyra matriser som anger risker ifrån olika händelser, se bilaga 1: *Struktursamling*, struktur *R*. Matriserna är av storlek *n*x1, där *n* är lika med antalet noder i topografin. Platserna i matriserna anger nodnummret i topografin. Med andra ord så anger värdet på plats *k* risken för nod *k*.

Struktur *StRes*

Strukturen *StRes* byggs upp under Monte Carlo-simuleringen och innehåller data som beskriver händelserna som äger rum under projektilernas färd. Matriserna i *StRes* byggs på eller modifieras genom hela MC-simuleringen. *StRes* används för att plotta olika fördelningar för att få en bättre bild av vad som inträffat under simuleringens gång. Strukturen *StRes* återfinns i bilaga 1: *Struktursamling*.

Tidsbestämddetonation

I det fallet att projektilen har en given timer, *dtimer*, briserar projektilen vid denna tidpunkt om denna tidpunkt nås i MC-simuleringen. Detta sköts via huvudprogrammet *mainMC*. Har tiden *dtimer* passerats används funktionen *getDtpos* (se avsnitt: *2.2.5.21*) för att finna briseringspunkten. Splitterområdet beräknas sedan med *calcFrgmRisc_k* (se avsnitt: 2.2.5.15).

mainMC skapar även matrisen *bncpoints* som används vid beräkning av passagerisken i funktionen *createPassPoints* (se avsnitt 2.2.5.18). Matrisen innehåller alla positioner där studs inträffat under projektilfärden.

Syntax:

```
[StRes, R, T, pdata, tdata, MCtime, trjts_fig] =
mainMC(NbrOfMC);
```

Indata:

NbrOfMC:	Skalär, anger antalet projektiler som ska simuleras.

Utdata:

StRes:	Struktur enligt bilaga 1.
R:	Riskstruktur, se bilaga 1.
T:	Topografistruktur, se bilaga 1.
pdata:	Projektilstruktur, se bilaga 1.
tdata:	Struktur för projektilbanan, se bilaga 1.

MCtime:	Skalär, anger medeltiden för en simulerad projektilbana.
trjts_fig:	”Handle” till figur innehållande de 20 första projektilbanorna.

2.2.5.2 editPData

Funktionen *editPData* skapar projektilstrukturen *pdata* som innehåller parametrarna listade i bilaga 1: *Struktursamling*, struktur *pdata*.

Syntax:

pdata = editPData

Utdata:

pdata:	Projektilstruktur, se bilaga 1.

2.2.5.3 editTopo

Skapar strukturen *T* som innehåller parametrar för bestämning av topografin. Strukturen *T* återfinns i bilaga 1: *Struktursamling*.

Syntax:

T = editTopo(pdata)

Indata:

pdata: Projektilstruktur, se bilaga 1.

Utdata:

T: Topografistruktur, se bilaga 1.

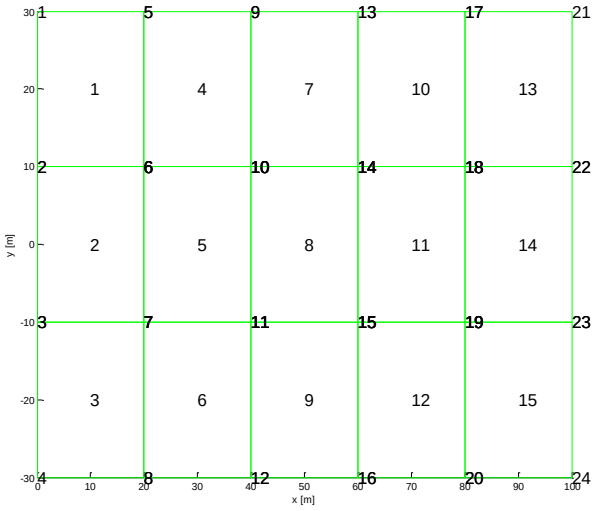
2.2.5.4 createtopografi4

Funktionen *createtopografi4* tar in strukturen *T* och modifierar dess innehåll samt utökar den med matriser och parametrar som bestämmer utseendet och egenskaperna för topografin. Strukturen *T* återfinns i bilaga 1.

Nedan följer en enkel beskrivning av matriserna i strukturen *T* med hjälp av ett exempel.

Topografin i figur 2.14 består av 15 element med *sz = 20m*. I topografin finns samtliga elementnummer och nodummer utritade. Elementen är numrerade 1-15 och är markerade med respektive nummer mitt i elementet. Noderna är numrerade 1-24 och är utritade i nodpunkterna. Topografin ligger i *xy*-planet och alla *z*-koordinater är noll.

För att illustrera hur de topografiska matriserna är konstruerade tar vi elementnummer 5 som exempel.



Figur 2.14, exempel på topografi med utmarkerade element- och nodnummer

Element 5 beskrivs av fyra koordinater. Detta är koordinaterna för noderna 6, 7, 10 och 11 som enligt bilden ligger i element 5:s hörn. Dessa koordinater kan tas fram på två olika sätt. Antingen tar man ut raderna 6, 7, 10 och 11 ur matrisen *coord* vilket resulterar i följande:

2.2.5.4.e1

$$coord(6,:) = [20,10,0]$$
$$coord(7,:) = [20,-10,0]$$
$$coord(10,:) = [40,10,0]$$
$$coord(11,:) = [40,-10,0]$$

eller så tar man ut rad 5 ur matriserna *Ex*, *Ey* och *Ez*.

2.2.5.4.e2

$$Ex(5,:) = [20,20,40,40]$$
$$Ey(5,:) = [10,-10,-10,10]$$
$$Ez(5,:) = [0,0,0,0]$$

Båda dessa anger samma positioner fast på två olika sätt. Matrisen *coord* anger direkt koordinaterna för en given nod medan matriserna *Ex*, *Ey* och *Ez* anger *x*-, *y*- och *z*-

koordinaterna för noderna tillhörande ett givet element. Användning av matriserna *Ex*, *Ey*, *Ez* anger även orienteringen av elementen som förklaras närmare i beskrivningen av matrisen *Edof* nedan.

Rad 5 ur *Edof*-matrisen för topografin i figur 2.14 innehåller följande:

2.2.5.4.e3

$$Edof\,(5,:) = [5,6,7,11,10]$$

Edof anger med första kolonnen vilket element det är som studeras och sedan följer de tillhörande noderna ordnade efter elementets orientering. Samtliga element är orienterade som element 5 i detta exempel. Första noden tillhörande ett element är alltså den övre vänstra noden i förhållande till figur 2.14 ovan. Orientering är sedan moturs. Detta ger för element 10:

2.2.5.4.e4

$$Edof\,(10,:) = [10,13,14,18,17]$$

Dessa matriser är komplement till varandra och används på skilda ställen i programmet. *Ex*, *Ey* och *Ez* används framförallt i plottningssammanhang medan *coord* och *Edof* används för att finna vilka noder som är av intresse när riskvärden ska beräknas samt för snabb och enkel sortering.

Syntax:

T = createTopografi4(T, pdata)

Indata:

T:	Topografistruktur, se bilaga 1.
pdata:	Projektilstruktur, se bilaga 1.

Utdata:

T:	Topografistruktur, se bilaga 1.

2.2.5.5 createWall

Målet definieras med hjälp av strukturen *target* som innehåller parametrar som anger målets bredd och höjd samt avstånd från avfyrningspositionen. Målet som skapas kan liknas vid en vägg med given bredd och höjd men som saknar djup. Strukturens parametrar finns listade i tabellen nedan.

Struktur target

Parameterbeteckning	Beskrivning	Enhet
b	Målets bredd i sidled vänster eller höger, total bredd=2b	m
h	Målets höjd	m
x	Avstånd i x-led till målet från avfyrningspositionen	m

Syntax:

target = createWall(b, h, x)

Indata:

b: Se tabell ovan.
h: Se tabell ovan.
x: Se tabell ovan.

Utdata:

target: Målstruktur med parametrar enligt tabellen ovan

2.2.5.6 editTData

Funktionen *editTData* skapar eller ändrar i banstrukturen *tdata* som innehåller parametrar för beskrivning av projektilbanan. Strukturen *tdata* återfinns i bilaga 1: *Struktursamling*. *editTData* används i två olika fall, dels vid avfyrning av projektil samt efter studs.

Syntax:

tdata = editTData(tdata, v, e, s)
tdata = editTData

Indata:

tdata: Projektilbandata före studseffekten

v: 1x2 matris innehållande projektilens fart före och efter studs

$v = [v_{före}\ v_{efter}]$

e: 1x2 matris innehållande projektilens nedslagsvinkel, e_i, samt elevationsvinkel, e_u, före respektive efter studs.

$e = [e_i\ e_u]$

s: 1x2 matris innehållande projektilens sidvinkel i förhållande till linjen y = 0 före och efter studs

$s = [s_{före}\ s_{efter}]$

Utdata:

tdata: Banstruktur med parametrar enligt bilaga 1.

OBS: Anges inte indata bestäms *tdata* utifrån avfyrningspositionen

2.2.5.7 getAim

Beräknar vilken elevationsvinkeln *e0* samt sidvinkel *s0* som behövs för att få direktträff av målet. Sidvinkeln är alltid noll eftersom målet placeras med centrum i $y = 0$ och avfyrningen sker i x-led.

Träffpunkten på målet väljs till $p_t = [x_t\ 0\ h/2]$, där x_t är avståndet till målet från origo och h är målets höjd.

För att finna rätt elevationsvinkel tar funktionen först fram två vinklar, en elevationsvinkel *a1* som resulterar i träffpunkten p_1 hitom p_t samt vinkeln *a2* som resulterar i träffpunkten p_2 som ligger bortom p_t. Dessa vinklar tas fram genom grov stegning från en låg elevationsvinkel tills dess att målet har överskjutits. Projektilbanorna beräknas med funktionen *xjob_trjectory_kcd2* som beskrivs i avsnitt 2.2.5.9. Detta resulterar i att $a1<a2$. Anledning till varför dessa vinklar behövs är för att begränsa området inom vilket den riktiga vinkeln finns. I ett allmänt fall behöver inte en större elevationsvinkel motsvara en längre räckvidd, men mellan vinklarna *a1* och *a2* har nu ett sådant område skapats.

En ny elevationsvinkel beräknas enligt

2.2.5.7.e1

$$e = \frac{a1 + a2}{2}$$

och på grund av hur *a1* och *a2* har bestämts ligger den nya slut positionen, p_e, ifrån projektilbanan beräknad med elevations vinkel e på avstånd mellan p_1 och p_2 ifrån avfyrningen. Funktionen *targethit* kontrollerar om målet har träffats (se avsnitt 2.2.5.12). Har målet missats jämförs slutpositionens x-koordinat, x_e med x_t och vinklarna modifieras enligt:

2.2.5.7.e2

$$x_e < x_t \Rightarrow a1 = e$$
$$x_e > x_t \Rightarrow a2 = e$$

Har målet träffats jämförs istället slutpositionens z-koordinat, z_e, med $h/2$ och vinklarna modifieras enligt:

2.2.7.7.e3

$$z_e < h/2 \Rightarrow a1 = e$$
$$z_e > h/2 \Rightarrow a2 = e$$

En ny elevationsvinkel beräknas enligt ekvation 2.2.5.7.e1 och proceduren görs om. Letandet efter rätt vinkel avslutas då träffpunkten p_t har träffats så när som på 1mm.

Syntax:

```
[e0, s0, count]  = getAim(tdata, target)
```

Indata:

tdata:	Struktur för projektilbanan, se bilaga 1.
target:	Målstruktur, se bilaga 1.

Utdata:

e0:	Elevationsvinkel för direktträff av mål.
s0:	Vinkel i sida för direktträff av mål (=0).
count:	Antal itereringar som krävdes för att finna korrekt vinkel.

2.2.5.8 devFiringAngles

Funktionen devierar siktet enligt bestämd sannolikhetsfördelning. *devFiringAngles* varierar siktet enligt tre olika alternativ:

1) Chi2-fördelad elevationsvinkel samt normalfördelad vinkel i sida.
2) Normalfördelade vinklar i både elevation och sida.
3) Ingen deviation i elevation och normalfördelad vinkel i sida
4) Chi2-fördelad elevationsvinkel och likformigtfördelad vinkel i sida

De nya vinklarna beräknas med MATLAB:s slumpgeneratorer *chi2rnd* [8], för chi2 fördelade vinklar, *normrnd* [8], för normalfördelade vinklar och *unifrnd* [8], för likformigtfördelade vinklar.

Chi2 fördelningen ges av (*Sannolikhetsteori och statistikteori med tillämpningar*, s. 154 [3]):

2.2.5.8.e1

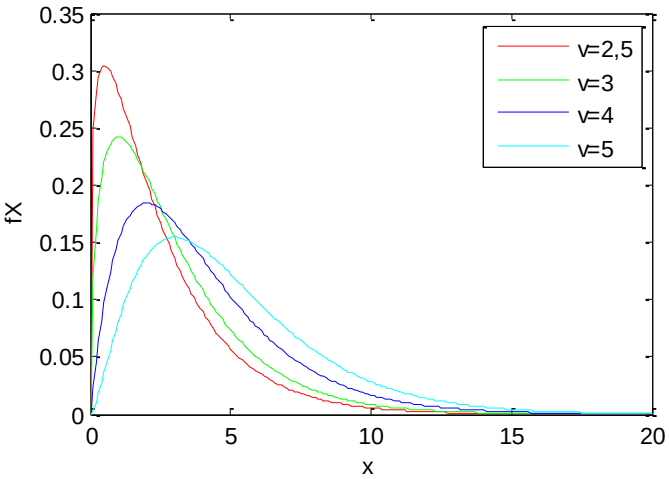
$$f_X(x) = \begin{cases} \dfrac{x^{\nu/2-1} \cdot e^{-x/2}}{\Gamma(\nu/2) \cdot 2^{\nu/2}}, & x > 0 \\ 0, & x \le 0 \end{cases}$$

Där ν är lika med antalet frihetsgrader. Gamma funktionen Γ definieras som (*Sannolikhetsteori och statistikteori med tillämpningar*, s. 64 [3]):

2.2.5.8.e2

$$\Gamma(c) = \int_0^\infty x^{c-1} \cdot e^{-x} dx, \qquad c > 0$$

Chi2 fördelningen används för att modellera att skytten har större sannolikhet att rikta fel uppåt än nedåt vid avfyrning med små elevationsvinklar. I figur 2.15 finns chi2 fördelningen representerad för några olika frihetsgrader.



Figur 2.15, chi2 fördelningar

Störst sannolikhet skall påträffas vid siktet som ger direktträff av mål, samtidigt som det skall vara möjligt för skytten att avfyra i negativa vinklar. Lösningarna till dessa problem redovisas nedan.

Derivation av ekvation 2.2.5.8.e1 med avseende på *x* ger:

2.2.5.8.e3

$$\frac{d}{dx}\left(\frac{x^{\nu/2-1}\cdot e^{-x/2}}{\Gamma(\nu/2)\cdot 2^{\nu/2}}\right) = \ldots = \frac{1}{2^{\nu/2}\cdot\Gamma(\nu/2)}\cdot\left(\frac{\nu-2}{2\cdot x}-\frac{1}{2}\right)\cdot x^{(\nu-2)/2}\cdot e^{-x/2}$$

Genom att ansätta detta uttryck till noll fås sedan samtliga maxima och minima till chi2-fördelningen.

2.2.5.8.e4

$$\frac{1}{2^{\nu/2}\cdot\Gamma(\nu/2)}\cdot\left(\frac{\nu-2}{2\cdot x}-\frac{1}{2}\right)\cdot x^{(\nu-2)/2}\cdot e^{-x/2} = 0 \Rightarrow \begin{cases} x = 0 \\ x = \infty \\ x = \nu - 2 \end{cases}$$

Där $x = 0$ och $x = \infty$ är minimum enligt figur 2.15, medan $x = \nu - 2$ är det enda maximumet. Värdet på den mest troliga vinkeln, x_0, i chi2 fördelningen är med andra ord lika med ν-2. Elevationsvinkeln, e_0, som resulterar i direktträff av mål ska ansättas lika med x_0, men för att kunna tillåta negativa vinklar behöver den mest troliga vinkeln definieras som:

2.2.5.8.e5

$x_0 = e_0 + \tau_-$, τ_- är lika med den tillåtna negativa vinkeln ifrån avfyrningen.

På så sätt kan den beräknade vinkeln ifrån chi2 fördelningen sedan subtraheras med τ_- för att åter vara tillbaka till $x_0 = e_0$. Samtidigt möjliggörs nu negativa elevationsvinklar.

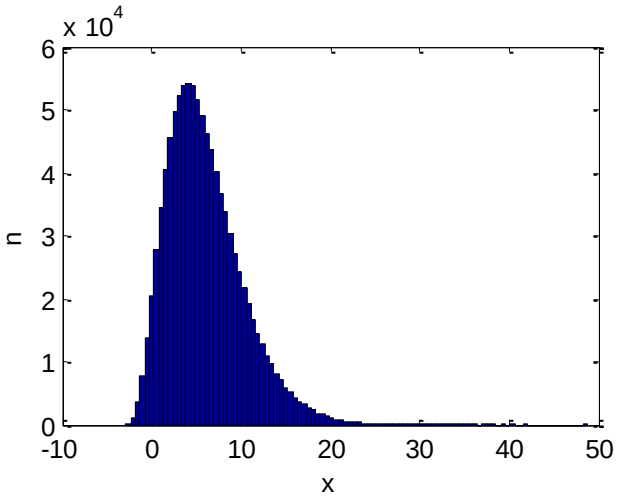
Antalet frihetsgrader ν kan beräknas utifrån resultatet av 2.2.5.8.e4 och 2.2.5.8.e5.

2.2.5.8.e6

$$\nu = e_0 + \tau_- + 2$$

Figur 2.16 illustrerar fördelningen för 10^6 chi2-fördelade vinklar med e_0 satt till 4 och τ_- lika med 3.

Denna fördelning bör endast andvändas vid relativt små e_0. Vid stora e_0 blir chi2-fördelningen väldigt bred och elevationsvinklarna från avfyrning bör devieras med normalfördelning för att få en realistisk spridning.



Figur 2.16, histogram över chi2-fördelade elevationsvinklar, $e_0 = 4$, $\tau_- = 3$.
x = vinklar (grader), n = antal vinklar

De normalfördelade samt likformigtfördelade vinklarna fås direkt med MATLAB funktionerna *normrnd* [8] respektive *unifrnd* [8] (För information om dessa fördelningar se *Sannolikhetsteori och statistikteori med tillämpningar*, s. 59 och 62 [3]):

Syntax:

[e, s] = devFiringAngles(e0, s0, devFangles)

Indata:

e0:	Elevationsvinkel från funktionen *getAim*, se avsnitt 2.2.5.7
s0:	Vinkel i sida (=0)
devFangles:	1x3 matris med parametrar för bestämning av sannolikhetsfördelning.

$devFangles = [edev, sdev, choise]$, *edev* är standardavvikelsen i elevation, *sdev* är standardavvikelsen i sida eller intervallgräns beroende på valet av *choise*. *choise* är en skalär, 1-4, som anger val av fördelningar enligt ovan. Vid likformigfördelning slumpas vinklarna mellan –*sdev* och *sdev*.

Utdata:

e:	Elevationsvinkel
s:	Vinkel i sida

2.2.5.9 xjob_trjectory_kCd_2

Beräknar projektilbana från given startposition och starttid till marknedslag med konstant luftmotståndskoefficient för de två fallen: med raketdrift, respektive utan raketdrift som beskrivs i teorin, avsnitt 2.2.1.1. Funktionen använder MATLAB:s inbyggda differentiallösare *ode23* [8] för att beräkna lösningar till differentialekvationen T2.e10. *xjob_trjectory_kCd_2* använder den konstanta luftmotståndskoefficienten *kCd* i strukturen *tdata* (se bilaga 1: S*truktursamling*). Accelerationen på projektilen ifrån luften ges av:

2.2.5.9.e1

$$a_{air} = -\frac{D}{m} = -kCd \cdot M^2$$

Där *D* ges av ekvation T2.e8 och *M* är machtalet enligt ekvation T2.e6. Parametern *kCd* fås ifrån utomstående testskjutningar via mätning med dopplerradar på projektilen. *xjob_trjectory_kCd_2* applicerar sedan raketdriften på projektilen utifrån parametrarna i understrukturen *rocket* (se bilaga 1: *Struktursamling*) i projektilens rörelseriktning enligt differentialekvation T2.e10.

Lösningen av differentialekvaiton T2.e10 ges av nya matriser som läggs till i strukturen *tdata* enligt tabellen nedan.

Tillägg till struktur tdata

Parameterbeteckning	Beskrivning	Värde och/eller enhet
start_dir	1x3 vektor som anger begynnelseriktningen för projektilbanan	Normerad riktningsvektor
t	nx1 matris som anger tidpunkterna då differentialekvation T2.e10 har lösts	s
p	nx3 matris som anger positionerna i de tidpunkter, t, då differentialekvation T2.e10 har lösts	m
v	nx3 matris som anger hastigheterna i de tidpunkter, t, då differentialekvation T2.e10 har lösts	m/s
absv	nx1 matris som anger farten i de tidpunkter, t, då differentialekvation T2.e10 har lösts	m/s
t_top	Anger tidpunkten då projektilbanan var på sin högsta höjd	s
p_top	1x3 matris som anger projektilens position vid tiden t_top	m
v_top	1x3 vektor som anger projektilens hastighet vid tiden t_top	m/s

Syntax:

tdata = xjob_trjectory_kCd_2(tdata)

Indata:

tdata: Struktur för projektilbanan, se bilaga 1.

Utdata:

tdata: Struktur för projektilbanan, se bilaga 1 samt tabellen ovan.

2.2.5.10 calcFrgmRisc_rN

Beräknar riskområdet som uppstår framför mynningen till vapnet vid avfyrning. Funktionen beräknar en vektor ***c*** enligt:

2.2.5.10.e1

$$\bar{c} = r \cdot [\cos(s), \sin(s)]$$

s = sidvinkeln vid avfyrning
r = riskavståndet framför vapnet

Denna vektor beskriver skjutriktningen i *xy*-planet. Därefter beräknas *non* stycken testvektorer, $\boldsymbol{tp_i}$, som sträcker sig ifrån avfyrningspunkten, ***pa***, till respektive nod enligt:

2.2.5.10.e2

$$\overline{tp_i} = \overline{coord}(i,:) - \overline{pa}$$

$$i = 1,2,...,non$$

(För beskrivning av matrisen *coord* se avsnitt 2.2.5.4)

Med skalärproduktens definition (*Linjär Algebra* s. 63 [6]) beräknas sedan vinkeln α mellan testvektorn och avfyrningsriktningen enligt:

2.2.5.10.e3

$$\bar{c} \cdot \overline{tp_i} = \left|\bar{c}\right| \cdot \left|\overline{tp_i}\right| \cdot \cos(\alpha_i) \Rightarrow$$

$$\Rightarrow \alpha_i = a\cos\left(\frac{\bar{c} \cdot \overline{tp_i}}{\left|\bar{c}\right| \cdot \left|\overline{tp_i}\right|}\right)$$

Om $\alpha_i \leq N$, N är riskvinkeln framför vapnet enligt teorin i avsnitt 2.2.1.5, kontrolleras längden, l_i, av testvektorn. Om denna längd är mindre eller lika med *r*, riskavståndet framför vapnet (se avsnitt 2.2.1.5), får nod *i* riskvärdet 1.

Syntax:

Res = calcFrgmRisc_rN(pdata, T, pos, s)
Res = calcFrgmRisc_rN(pdata, T, tdata)

Indata:

pdata:	Struktur för projektil, se bilaga 1.
T:	Topografistruktur, se bilaga 1.
pos:	Avfyrningsposition, pos = [*x*, *y*, *z*]
s:	Sidvinkeln vid avfyrning
tdata:	Struktur som beskriver projektilbanan, se bilaga 1.

Utdata:

Res:	*n*x1 matris, *n* = *non*, som innehåller riskvärdena 0 eller 1.

2.2.5.11 calcFrgmRisc_Nu

Beräknar riskområdet som uppstår bakom vapnet vid avfyrning. Arbetar på samma sätt som *calcFrgmRisc_rN* (se avsnitt 2.2.5.10) med skillnaden att vektorn ***c*** beräknas enligt:

2.2.5.11.e1

$$\bar{c} = u \cdot [\cos(s + \pi), \sin(s + \pi)]$$

s = sidvinkeln vid avfyrning
u = riskavståndet bakom vapnet

Testvinklarna α_i som beräknas enligt ekvation 2.2.5.10.e2 samt 2.2.5.10.e3 jämförs sedan med riskvinkeln *Nu* och längderna av testvektorerna från ekvation 2.2.5.10.e2 jämförs med riskavståndet *u* (se teorin i avsnitt 2.2.1.5).

Syntax:

Res = calcFrgmRisc_Nu(pdata, T, pos, s)
Res = calcFrgmRisc_Nu(pdata, T, tdata)

Indata:

pdata:	Projektilstruktur, se bilaga 1.
T:	Topografistruktur, se bilaga 1.
pos:	Avfyrningsposition, *pos* = [*x*, *y*, *z*]
s:	Vinkel i sida vid avfyrning
tdata:	Struktur som beskriver projektilbanan, se bilaga 1.

Utdata:

Res:	*n*x1 matris, *n* = *non*, som innehåller riskvärdena 0 eller 1.

2.2.5.12 targethit

En projektilbana sträcker sig ursprungligen ifrån avfyrningspositionen till marknedslag. Funktionen *targethit* kontrollerar om projektilen har träffat målet genom att undersöka om banan passerar igenom målets utbredning i rummet och kortar projektilbanan till träffpunkten. Detta löses med ett par logiska uttryck.

Målet består av parametrarna (enligt bilaga 1, struktur *target*):

tx: avståndet till målet i *x*-led från avfyrningen (origo)
b: målets utsträckning i *y*-led
h: målets höjd

targethit kontrollerar först om projektilbanan har passerat *tx* positionen i *x*-led. Är detta sant beräknas *y*- respektive *z*-koordinaten för projektilbanan i punkten $x = tx$, *ty* respektive *tz*. Detta görs med hjälp av MATLAB:s inbyggda interpolationsalgoritm *interp1* [8]. Sedan kontrolleras följande villkor:

2.2.5.12.e1

$$-b \le ty \le b$$

$$tz \le h$$

Uppfylls samtliga villkor har projektilen träffat målet. Banstrukturen *tdata*:s (se bilaga 1) matriser modifieras så att de endast sträcker sig fram till träffpunkten. Matriserna som förkortas listas nedan:

Modifierade matriser i struktur tdata

Parameterbeteckning	Beskrivning	Enhet
t	nx1 matris som anger tidpunkterna då differentialekvationen har lösts, förkortad till briseringstiden	s
p	nx3 matris som anger positionerna i de tidpunkter, t, då differentialekvationen har lösts, förkortad till briseringspunkten	m
v	nx3 matris som anger hastigheterna i de tidpunkter, t, då differentialekvationen har lösts, förkortad till briseringstiden	m/s
absv	nx1 matris som anger farten i de tidpunkter, t, då differentialekvationen har lösts, förkortad till briseringstiden	m/s

Syntax:

```
[hit, tdata] = targethit(tdata, target)
```

Indata:

tdata: Banstruktur, se bilaga 1.
target: Målstruktur, se bilaga 1.

Utdata:

hit:	Boolean, hit = *sant* om träff av mål, hit = *falskt* annars.
tdata:	Banstruktur, se bilaga 1 samt tabell ovan.

2.2.5.13 trj_detonation

Kontrollerar om en banbrisad inträffat på grund utav konstruktionsfel. Risken att detta händer är binomialfördelat med två olika sannolikheter (se bilaga 1, struktur *pdata*, parametrar: *before_armingP* respektive *after_armingP*).

Banbrisad kontrolleras med hjälp av MATLAB:s *random number generator*, *binornd* [8]. Har en banbrisad inträffat förkortas projektilbanan till brisadpunkten.

Var brisadpunkten befinner sig är uträknat ifrån en likformigfördelning i tiden med MATLAB:s *random number generator*, *unifrnd* [8]. Sker banbrisaden före armering ligger tidpunkten för brisaden mellan tiden t_0, avfyrningstiden, och tiden t_a, armeringstiden. Sker banbrisaden efter armering ligger briseringstidpunkten mellan tiden t_a och tiden t_n, tidpunkten för marknedslag. Detta resulterar i att även om en projektilbana får sannolikheten 1 för banbrisad kan denna händelse försvinna om samma projektil träffar målet före brisadtidpunkten.

Matriserna i tdata som modifieras då banbrisad inträffar är listade i tabellen nedan.

Modifierade matriser i struktur tdata

Parameterbeteckning	Beskrivning	Enhet
t	nx1 matris som anger tidpunkterna då differentialekvationen har lösts förkortad till briseringstiden	s
p	nx3 matris som anger positionerna i de tidpunkter, t, då differentialekvationen har lösts, förkortad till briseringpunkten	m
v	nx3 matris som anger hastigheterna i de tidpunkter, t, då differentialekvationen har lösts, förkortad till briseringstiden	m/s
absv	nx1 matris som anger farten i de tidpunkter, t, då differentialekvationen har lösts, förkortad till briseringstiden	m/s

Syntax:

[tdata, trj_deto] = trj_detonation(pdata, tdata)

Indata:

pdata: Projektilstruktur, se bilaga 1.
tdata: Banstruktur, se bilaga 1.

Utdata:

tdata: Modifierad banstruktur, se bilaga 1 samt tabellen ovan.
trj_deto: Boolean, trj_deto = *sant* om banbrisad inträffat, trj_deto = *falskt* annars.

2.2.5.14 RiscAreaTargetHit

Beräknar briseringspositionerna som uppkommer vid direktträff av målet. Dessa koordinater används sedan i *calcFrgmRisc_k* (se 2.2.5.15) som beräknar riskområdet.

Vid träff av mål förkortas projektilbanan till träffpunkten i funktionen *targethit* (se avsnitt 2.2.5.12). *RiscAreaTargetHit* kontrollerar antalet projektilladdningar samt aktiveringstiderna hos laddningarna och beräknar briseringspositioner med hjälp av dessa tider. Till detta används funktionen *getDtpos* (se avsnitt 2.2.5.20) som beräknar koordinaten för projektilen vid en viss tid.

Briseringstidpunkten, t_b, fås enligt:

2.2.5.14.e1

$$t_b = t_{hit} + t_a$$

där t_{hit} är tidpunkten då projektilen träffade målet och t_a är aktiveringstiden för projektilen. Tiden t_b skickas sedan till *getDtpos* som beräknar positionen $\boldsymbol{p_b}$. Projektilen förflyttas på så sätt en sträcka motsvarande tiden t_a framåt i sin ursprungliga projektilbana som om målet inte funnits. Detta är för att simulera inverkan av projektiler med förpenetrator. Den första laddningen spränger ett hål i väggen som den andra laddningen sedan kan passera igenom.

Syntax:

Res = RiscAreaTargetHit(tdata, tdata0, pdata, T, target)

Indata:

tdata:	Banstruktur från *targethit*, se avsnitt 2.2.5.12.
tdata0:	Ursprunglig banstruktur från *xjob_trjectory_kcd2*, se bilaga 1.
pdata:	Projektilstruktur, se bilaga 1.
T:	Topografistruktur, se bilaga 1.
target:	Målstruktur, se bilaga 1.

Utdata:

Res:	*n*x1 matris, *n* = *non*, som innehåller riskvärdena för splitter, 0-1.

2.2.5.15 calcFrgmRisc_k

Beräknar splitterutbredningen vid brisad i given briseringspunkt, $\boldsymbol{p_b}$, med hjälp av riskavståndet *k* i enlighet med ekvation T2.e23 i teorin, avsnitt 2.2.1.6.

Funktionen skapar testvektorerna $\boldsymbol{tp_i}$ enligt:

2.2.5.15.e1

$$\overline{tp_i} = \overline{coord}(i,:) - \overline{p_b}$$
$$i = 1,2,...,non$$

Testvektor *i* tillhör på så sätt nod *i* (se avsnitt 2.2.5.4 för beskrivning av *coord*). De testvektorer som har längd mindre eller lika med *k* tilldelas sedan riskvärden utifrån splitterfördelningen nedan.

2.2.5.15.e2

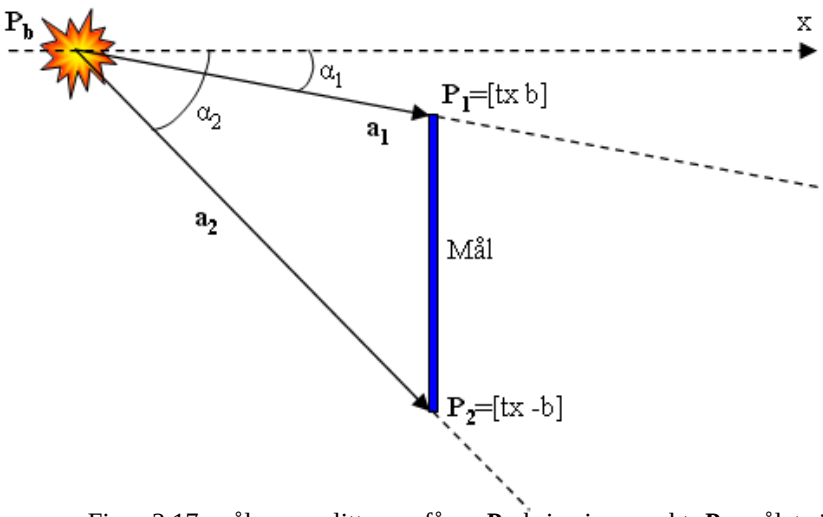
$$r_i = 1 - \frac{\left|\overline{t_{pi}}\right|^2}{k^2}$$
$$i = 1,2,...,non$$

Splitterrisken r_i för nod *i* blir ett värde mellan 0 och 1, 1 för noden i briseringspunkten samt 0 för noderna på avstånd större än *k* radiellt ut från briseringspunkten.

Målet som splitteruppfång

Målet som skapas i topografin kan fungera som splitteruppfång. Detta löses genom att finna de noder som ligger bakom målet i förhållande till brisadpunkten och ansätta riskvärdet 0 till dessa.

Från briseringspunkten till målets ändpunkter skapas två vektorer, $\boldsymbol{a_1}$ och $\boldsymbol{a_2}$ enligt figur 2.17 nedan.



Figur 2.17, mål som splitteruppfång, $\boldsymbol{P_b}$: briseringspunkt, $\boldsymbol{P_1}$: målets övre ändpunkt (se avsnitt 2.2.5.5), $\boldsymbol{P_2}$: målets nedre ändpunkt (se avsnitt 2.2.5.5)

Vektorerna $\boldsymbol{a_1}$ och $\boldsymbol{a_2}$ ges av:

2.2.5.15.e3

$$\overline{a}_i = \overline{P}_i - \overline{P}_b$$
$$i = 1,2$$

Vinklarna α_1 samt α_2 fås med definitionen av skalärprodukt enligt:

2.2.5.15.e4

$$\alpha_i = \arccos(\frac{a_i \cdot n}{|a_i|})$$
$$n = [1,0,0]$$
$$i = 1,2$$

Härefter skapas testvinklar, $t\alpha_i$, med testvektorerna $\boldsymbol{tp}_i$ på motsvarande sätt som ekvation 2.2.5.15.e4.

2.2.5.15.e5

$$t\alpha_i = \arccos(\frac{tp_i \cdot n}{|tp_i|})$$
$$n = [1,0,0]$$
$$i = 1,2,...non$$

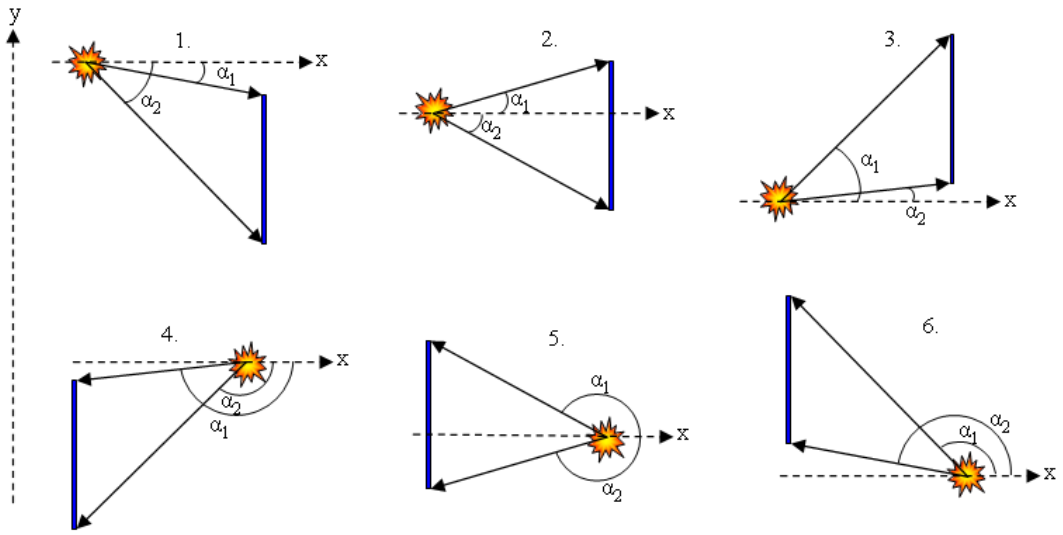
De nodrisker som ska ansättas till noll representeras av de testvinklar som ligger inom intervallet:

2.2.5.15.e6

$$\alpha_1 \leq t\alpha_i \leq \alpha_2$$
$$i = 1,2,...n$$

och som samtidigt har en *x*-koordinaten som ligger bortanför målet. Nodrisker inom intervallet i utryck 2.2.5.15.e6 men hitom målet sett från briseringspunkten nollställs inte.

Eftersom att cosinus är en periodisk funktion samt att briseringar kan ske på båda sidor om målet uppstår sex olika fall beroende på var briseringspunkten befinner sig, se figur 2.18 nedan.



Figur 2.18, olika fall vid brisering med splitteruppfång

Villkoret i 2.2.5.15.e6 gäller endast för fall 6 i figur 2.18. Liknande villkor fastställs för samtliga fall. Motsvarande problem uppstår även när testvinklarna skapas. För att få en ökning av testvinklarna från 0 till 2π ändras alla testvinklar tillhörande testvektorer med negativ *y*-riktning enligt:

2.2.5.15.e7

$$\alpha t_i = 2\pi - \alpha t_i$$

Då briseringar sker på höjd över målets höjd, *h*, försvinner effekten av målet som splitteruppfång och hela det cirkulära splitterområdet returneras. Det samma gäller för briseringar *tillräckligt* nära målet. Då briseringar sker *tillräckligt* nära målet tolkas detta som brisering i målet. Denna tolkning behövs eftersom att målet saknar djup och har på så sätt igen volym (se bilaga 1, struktur *target*). På grund utav det kan i själva verket inte briseringar äga rum inuti målet. Innebörden av *tillräckligt* nära kan variera beroende på vilket fall som ska studeras.

Syntax:

Res = calcFrgmRisc_k(pos, k, T, target)

Indata:

pos:	1x3 matris, briserings koordinater i *x*-, *y*- och *z*-led.
k:	*k*-värdet för projektilen.
T:	Topografistruktur, se bilaga 1.
target:	Målstruktur, se bilaga 1.

Utdata:

Res:	*n*x1 matris, *n* = *non*, som innehåller riskvärdena för splitter, 0-1

2.2.5.16 Studs

Nedan finns två funktioner för beräkning av studseffekter enligt teorin i avsnitt 2.2.1.2. Skillnaden mellan dessa är hur de framställer deviationer i markplan.

2.2.5.16.1 calcBounce_normP2

Funktionen beräknar studsvinklar och hastigheter genom att normalfördela infallsvinkeln i elevation och sida enligt:

2.2.5.16.e1

$$e_i \in N(e_{i0}, ed)$$
$$s_i \in N(s_{i0}, sd)$$

e_{i0} och s_{i0} är de framräknade infallsvinklarna i elevation och sida ifrån projektilbanan. *ed* och *sd* är standardavvikelseparametrar för att beskriva terrängens ojämnhet. Dessa ansätts när topografin skapas (se avsnitt 2.2.5.3). De normal- samt likformigtfördelade vinklarna beräknas med MATLAB funktionerna *normrnd* respektive *unifrnd* [8].

e_{i0} och s_{i0} beräknas enligt:

2.2.5.16.e2

$$e_{i0} = \arctan\left(\frac{\left|\bar{v}_{in0}\right|}{\left|\bar{v}_{it0}\right|}\right)$$

$$s_{i0} = \arccos\left(\frac{\left|\bar{v}_{ix0}\right|}{\left|\bar{v}_{it0}\right|}\right)$$

Där $\boldsymbol{v_{i0}}$, $\boldsymbol{v_{in0}}$ och $\boldsymbol{v_{it0}}$ är den inkommande hastigheten, den inkommande hastighetens normalkomposant i förhållande till planet med normalen [*0, 0, 1*] samt den tangentiella hastigheten i *xy*-planet. $\boldsymbol{v_{iox}}$ är *x*-komposanten av den inkommande hastigheten.

Nya infallshastigheter beräknas utifrån de normalfördelade vinklarna i ekvation 2.2.5.16.e1 enligt:

2.2.5.16.e3

$$\bar{v}_i = \left|\bar{v}_{i0}\right| \cdot \left[\cos(s_i)\cdot\cos(e_i), \sin(s_i)\cdot\cos(e_i), \sin(e_i)\right]$$
$$\bar{v}_{in} = \bar{v}_i \cdot \bar{n}, \quad \bar{n} = [0,0,1]$$
$$\bar{v}_{it} = \bar{v}_i - \bar{v}_{in}$$

Studsparametrarna α och β beräknas sedan med funktionen *calcAlphaBeta* (se avsnitt 2.2.5.24) och de utgående hastigheterna beräknas enligt:

2.2.5.16.e4

$$\bar{v}_{un} = -\beta \cdot \bar{v}_{in}$$
$$\bar{v}_{ut} = \alpha \cdot \bar{v}_{it}$$
$$\bar{v}_{u} = \bar{v}_{un} + \bar{v}_{ut}$$

Den utgående vinkeln i elevation beräknas tillsist enligt:

2.2.5.16.e5

$$e_u = \arcsin\left(\frac{\bar{v}_u \cdot \bar{n}}{\left|\bar{v}_u\right|}\right)$$

Utfallsvinkeln i sida behöver inte beräknas och kommer att vara lika med den normalfördelade infallsvinkeln i sida eftersom denna endast påverkas av hastighetsändringar i *xy*-planet och denna ändring är lika för *x*- och *y*-komposanterna eftersom att båda skalas med samma parameter α.

Syntax:

[V, E, S] = clacBounce_normP2(vi0, elmnbr, T, pdata)

Indata:

vi0:	1x3 vektor, infallandehastighet.
elmnbr:	Elementnumret för det element där studsen sker.
T:	Topografistruktur, se bilaga 1.
pdata:	Projektilstruktur, se bilaga 1.

Utdata:

V: 1x2 matris, beloppet av den inkommande respektive utgående hastigheten.

V = [v_i v_u]

E: 1x2 matris, infallsvinkel i elevation samt utfallsvinkel i elevation.

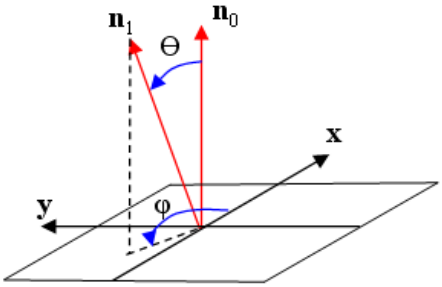
E = [e_i e_u]

S: 1x2 matris, infallsvinkel i sida samt utfallsvinkel i sida.

S = [s_i s_u]

2.2.5.16.2 calcBounce_normPn

Funktionen beräknar studsvinklar och hastigheter enligt teorin i avsnitt 2.2.1.2 genom att variera studsytans normalriktning. Topografins verkliga normal är $\boldsymbol{n}_0$=[*0, 0, 1*]. *calcBounce_normPn* beräknar en ny normal genom att ta ut en normalfördelad vinkel Θ med standardavvikelse *thetadev* (se bilaga 1, struktur *T*, parameter *thetadev*) samt en likformigtfördelad vinkel φ enligt:



Figur 2.19, slumpad normal till studsytan

2.2.5.16.e6

$$\Theta \in N(0, Thetadev)$$

$$\varphi \in U(0, 2\pi)$$

Dessa vinklar beräknas med MATLAB-funktionerna *normrnd* samt *unifrnd* [8].

En ny normal $\boldsymbol{n_1}$ (se figur 2.19) beräknas enligt:

2.2.5.16.e7

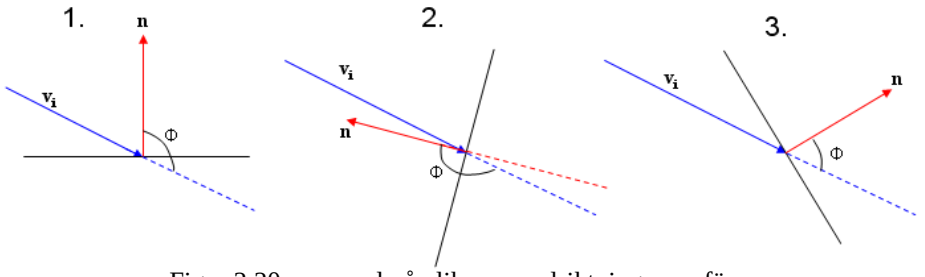
$$\bar{n}_1 = \left[\cos(\varphi)\cdot\sin(\Theta), \sin(\varphi)\cdot\sin(\Theta), \cos(\Theta)\right]$$

$\boldsymbol{v_{in}}$ och $\boldsymbol{v_{it}}$ beräknas sedan enligt ekvation 2.2.5.16.e3 med $\boldsymbol{n}$ = $\boldsymbol{n_1}$. Den utgående hastigheten och elevationsvinkeln beräknas med ekvation 2.2.5.16.e4 samt 2.2.5.16.e5. Jämfört med *calcBounce_normP2*, avsnitt 2.2.5.16.1, behöver även en ny vinkel i sida beräknas. Denna ges av:

2.2.5.16.e8

$$s_u = \arccos\left(\frac{\bar{v}_u \cdot \bar{n}}{|\bar{v}_u|}\right), \bar{n} = [1,0,0]$$

Med denna metod kan det uppstå ”overkliga” händelser. Vid en studs mot en yta kan inte vinkeln mellan den inkommande hastigheten och ytans normal vara mindre än π/2 enligt figur 2.20 nedan:



Figur 2.20, exempel på olika normalriktningar, *n*, för given infallshastighet, v_i.

Fall 3 i figuren ovan är exempel på en händelse som inte kan inträffa, projektilen kommer från undersidan av planet. *calcBounce_normPn* löser detta genom att anta att

en slumpad normal som ger resultat enligt fall 3 stoppar projektilen i nedslagspunkten. Vinkeln Φ i figur 2.20 beräknas enligt definitionen av skalärprodukt (*Linjär Algebra s. 63* [6]):

2.2.5.16.e9

$$\Phi = \arccos\left(\frac{\bar{v}_i \cdot \bar{n}}{\left|\bar{v}_i\right|}\right)$$

Syntax:

[V, E, S] = clacBounce_normPn(vi0, elmnbr, T, pdata)

Indata:

vi0:	1x3 vektor, infallshastigheten.
elmnbr:	Elementnumret för det element där studsen sker.
T:	Topografistruktur, se bilaga 1.
pdata:	Projektilstruktur, se bilaga 1.

Utdata:

V: 1x2 matris, beloppet av den inkommande respektive utgående hastigheten.

V = [v_i v_u]

E: 1x2 matris, infallsvinkel i elevation samt utfallsvinkel i elevation.

E = [e_i e_u]

S: 1x2 matris, infallsvinkel i sida samt utfallsvinkel i sida.

S = [s_i s_u]

2.2.5.17 bnc_type

Funktionen avgör vilken typ av studs som skett. Här finns tre fall:

1. Hastigheten efter studs är noll, projektilen har stannat/fastnat.
2. Studsen resulterar i brisering, två fall:
    a. Aktiveringstalet för projektilen (se bilaga 1, struktur *pdata*) är mindre än förhållande mellan nedslagsvinkel och markens hårdhet och projektilen har given aktiveringstid/tider.
    b. Samma som a, men projektilens aktiveringstid är lika med noll.
3. Projektilen aktiveras ej i stöten och en ny bana ska beräknas.

Funktionen kontrollerar vilket av händelserna som inträffat och returnerar rätt fall.

Syntax:

Type = bnc_type(impactangle, pdata, T, elmNbr, v)

Indata:

impactangle:	Nedslagsvinkel
pdata:	Projektilstruktur, se bilaga 1.
T:	Topografistruktur, se bilaga 1.
elmNbr:	Elementnummer där studs har inträffat.
v:	Projektilens fart efter studs.

Utdata:

Type:	Skalär, 1,2 eller 3, anger vilket fall som har inträffat.

2.2.5.18 createPassPoints

Den numeriska lösningen till rörelseekvationen T2.e10 i teoriavsnitt 2.2.1.1 består av en mängd koordinater för projektilbanan. För att använda Lagrange interpolationen som beskrivs i teorin, avsnitt 2.2.1.4, är det bra om projektilbanans koordinater är jämnt fördelade över *xy*-planet. Är inte punkterna jämnt fördelade eller för glest fördelade kan vissa noder få felaktiga riskvärden för passage.

createPassPoints tar in den ursprungliga projektilbanan och delar upp den i stycken där ett stycke sträcker sig från en studsposition (första stycket sträcker sig från avfyrningen) till nästa marknedslag (sista stycket kan avslutas över mark) och fördelar sedan koordinaterna i *xy*-planet så att det ligger på inbördes lika avstånd. Detta görs genom att approximera en rät linje över ett stycke och sedan ta ut jämnt fördelade koordinater. Approximeringen av linjen görs med MATLAB funktionen *polyfit* [8].

Syntax:

passpoints = createPassPoints(p, bncpoints, stepsize, tdata)

Indata:

p:	*n*x3 matris innehållande koordinaterna för projektilbanan, se bilaga 1, struktur *tdata*, parameter *p*.
bncpoints:	Slutpositioner för projektilbanans olika stycken, skapas i *mainMC*, se avsnitt 2.2.5.1
stepsize:	Önskat avstånd i *xy*-planet mellan de nya punkterna

Utdata:

passpoints:	*m*x3 matris med jämnt fördelade bankoordinater i förhållande till *xy*-planet, där *m* kan vara större eller mindre än *n* beroende på val av *stepsize*.

2.2.5.19 calcPassRisc1_1

Beräknar passagerisken enligt teorin i avsnitt 2.2.1.4. *calcPassRisc1_1* stegar igenom *passpoints* (se avsnitt 2.2.5.18) och beräknar elementnummer för samtliga koordinater med funktionen *getElmNbr* (se avsnitt 2.2.5.20). Koordinaterna samt elementnumrena skickas sedan till funktionen *formfunc* (se avsnitt 2.2.5.22) som interpolerar ut risken för passage till elementnoderna. Den lokala risken för elementet assembleras sedan in i den globala riskmatrisen med funktionen *insertRisc* (se avsnitt 2.2.5.23).

Syntax:

Rpass = calcPassRisc1_1(T, p, pdata)

Indata:

T:	Topografistruktur, se bilaga 1.
p:	*n*x3 matris med koordinater för projektilbanan (*passpoints*).
pdata:	Projektilstruktur, se bilaga 1.

Utdata:

Rpass:	nx1 matris, n = antal noder, innehållande risken för passage med värden från 0 till 1 enligt teorin i avsnitt 2.2.1.4.

2.2.5.20 getElmNbr

Beräknar elementnummer för en koordinat, ***p*** = [*x y*], i *xy*-planet med hjälp av topografimatriserna *Ex* och *Ey* (se avsnitt 2.2.5.4).

För att beskriva hur detta går till används följande exempel:

I figur 2.21 finns en koordinat ***p*** = [*65,5*], utmarkerad. Funktionen finner först det minsta avståndet *Dy* till de horisontella linjerna enligt:

2.2.5.20.e1

$$dy_i = |Ey(i,1) - y| + |Ey(i,2) - y|$$

$$i = 1...nelm$$

$$Dy = \min(dy_i)$$

I figur 2.21 b är dy_i:s två komponenter utritade för *i* = 10, 11 och 12.

Därefter beräknar funktionen vilka *i* som ger dy_i = *Dy*. I detta exempel resulterar detta i element nummer *i* = 2, 5, 8, 11 och 14.

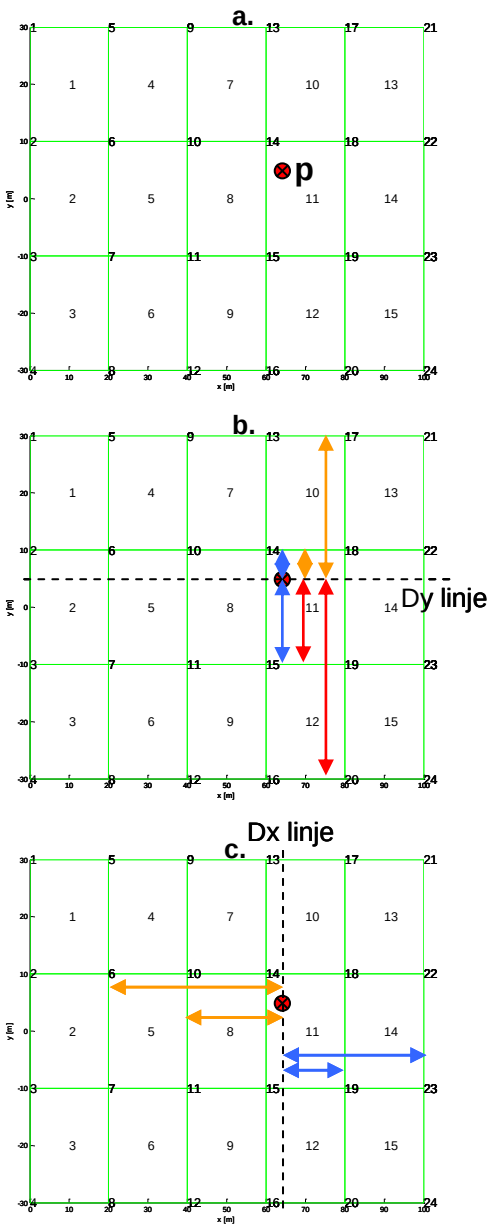
Sedan beräknas motsvarande avstånd *Dx* (avståndet till de vertikala linjerna) enligt:

2.2.5.20.e2

$$dx_j = |Ex(i,1) - x| + |Ey(i,3) - x|$$

$$j = 2,5,8,11,14$$

$$Dx = \min(dx_j)$$



Figur 2.21, exempel på beräkning av elementnummer. a) koordinat **p** i element som söks, b) dy_i komposanter till element 11 (blåa), 10 (gula) och 12 (röda), c) dx_i komposanter till element 5 (gula) och 14 (blåa)

I figur 2.21 c är dx_i:s komponenter utritade för *j* = 5 och 14. Det minsta *Dx*-värdet fås för *j* = 11 och det sökta elementet har funnits.

Om punkten ligger på en skärningslinje mellan två element returneras båda elementnumren och i fall att punkten ligger i skärningen mellan två linjer returneras samtliga fyra angränsande elementnummer.

Syntax:

elmNbr = getElmNbr(pos, T)

Indata:

pos:	1x3 matris med koordinaten i det sökta elementet
T:	Topografistruktur, se bilaga 1.

Utdata:

elmNbr:	1x*n* matris, *n* = 1,2 eller 4, med elementnummer

2.2.5.21 getDtpos

Beräknar briseringsposition i given projektilbana vid given tid t_b. Funktionen interpolerar fram *x*, *y* och *z* koordinaterna för tidpunkten t_b med hjälp av MATLAB-funktionen *interp1* [8]. Därefter förkortas projektilbanan till denna koordinat. Matriser i *tdata* (se bilaga 1) som förkortas finns listade i tabellen nedan.

Modifierade matriser i struktur tdata

Parameterbeteckning	Beskrivning	Enhet
t	nx1 matris som anger tidpunkterna då differentialekvation T2.e10 har lösts, förkortad till tiden t_b	s
p	nx3 matris som anger positionerna i de tidpunkter, t, då differentialekvation T2.e10 har lösts, förkortad till tiden t_b	m
v	nx3 matris som anger hastigheterna i de tidpunkter, t, då differentialekvation T2.e10 har lösts, förkortad till tiden t_b	m/s
absv	nx1 matris som anger farten i de tidpunkter, t, då differentialekvation T2.e10 har lösts, förkortad till tiden t_b	m/s

Syntax:

[pos, tdata] = getDtpos(tdata, time)

Indata:

tdata: Banstruktur, se bilaga 1.
time: Tidpunkten då projektilen befinner sig vid den sökta positionen.

Utdata:

pos: 1x3 matris med koordinaten för projektilen vid tiden *time*.
tdata: Modifierad banstruktur, se bilaga 1 samt tabellen ovan.

2.2.5.22 formfunc

Funktionen beräknar passagerisken över ett element för en given koordinat med Lagrange interpolation enligt teorin i avsnitt 2.2.1.4.

Syntax:

Ne = formfunc(ex, ey, p)

Indata:

ex:	1x4 matris med *x*-koordinater för givet elements noder med korrekt orientering (se avsnitt 2.2.5.4 samt bilaga 1, struktur *T*, parameter *Ex*).
ey:	1x4 matris med y-koordinater för givet element noder med korrekt orientering (se avsnitt 2.2.5.4 samt bilaga 1, struktur *T*, parameter *Ey*).
p:	1x2 eller 1x3 matris, koordinaten för projektilen i antingen *xy*-planet eller rummet.

Utdata:

Ne:	4x1 matris med den interpolerade risken för respektive elementnod.

2.2.5.23 insertRisc

Funktionen assemblerar in den lokala risken *Ne*, ifrån funktionen *formfunc* (se avsnitt 2.2.5.22), över ett element *j* till den globala riskmatrisen med hjälp av topografimatrisen *Edof* (se 2.2.5.4 samt bilaga 1, struktur *T*). Kolonn 2 till 5 i rad *j* i *Edof*-matrisen anger vilka noder som ska få den givna risken *Ne*. Dessa nodnummer är även radnummer i den globala riskmatrisen.

Riskvärdet för passage i en nod, för en projektilbana, antar endast det största värdet interpolationen från *formfunc* levererar. Funktionen *insertRisc* kontrollerar först om riskvärdet för passage i den globala riskmatrisen är större eller mindre än riskvärdet i *Ne* innan den assemblerar risken.

Syntax:

Risc = insertRisc(Ne, risc, T, ElmNbr)

Indata:

Ne:	4x1 lokal riskmatris.
risc:	*n*x1 global riskmatris, *n* = antalet noder.
T:	Topografistruktur, se bilaga 1.
ElmNbr:	Elementnummer

Utdata:

Risc:	*n*x1 global riskmatris, *n* = antalet noder.

2.2.5.24 calcAlphaBeta

Beräkningsmetoder för funktionen saknas på grund utav brist på utomstående simuleringar för framtagning av förhållanden enligt teorin i avsnitt 2.2.1.2. Funktionen *calcAlphaBeta* anger endast förbestämda värden för studsparametrarna α och β som ansätts direkt i funktionen.

Syntax:

[alpha, beta] = calcAlphaBeta(v, pdata, T, elmNbr)

Indata:

v:	1x3 matris med infallshastigheten
pdata:	Projektilstruktur, se bilaga 1.
T:	Topografistruktur, se bilaga 1.
elmNbr:	Elementnummer

Utdata:

alpha:	Studsparameter som anger förhållandet mellan de tangentiella hastigheterna enligt teorin i avsnitt 2.2.1.2.
beta:	Studsparameter som anger förhållandet mellan de normalriktade hastigheterna enligt teorin i avsnitt 2.2.1.2.

OBS: Indata till *calcAlphaBeta* är exempel på parametrar som kan vara av intresse vid eventuell fastställande av funktioner för beräkning av studsparametrarna.

2.2.5.25 mergetdata

Funktionen assemblerar ihop två *tdata* strukturer, *t1* samt *t2*, till en gemensam struktur. Detta är aktuellt efter studseffekt då ny bana beräknas utifrån nedslagspositionen.

I tabellen nedan listas de matriser i *tdata* strukturerna *t1* (före studs) samt *t2* (efter studs) som assembleras ihop vid studs.

Matriser i struktur t1 samt t2 som summeras

Parameterbeteckning	Beskrivning	Enhet
t	nx1 matris som anger tidpunkterna då differentialekvationen har lösts.	s
p	nx3 matris som anger positionerna i de tidpunkter, t, då differentialekvationen har lösts.	m
v	nx3 matris som anger hastigheterna i de tidpunkter, t, då differentialekvationen har lösts.	m/s
absv	nx1 matris som anger farten i de tidpunkter, t, då differentialekvationen har lösts.	m/s

mergetdata kontrollerar även om den högsta höjden som nåtts i *t2*, *p_top* (se bilaga 1, struktur *tdata*, parameter *p_top*), är större än den högsta höjden som nåtts före studs så att den returnerade strukturen innehåller de korrekta parametrarna: *p_top*, *t_top* och *v_top* vilka anger den högsta höjden, tiden för den högsta höjden samt hastigheten vid den högsta höjden för den totala projektilbanan.

Syntax:

tdata = mergetdata(t1, t2)

Indata:

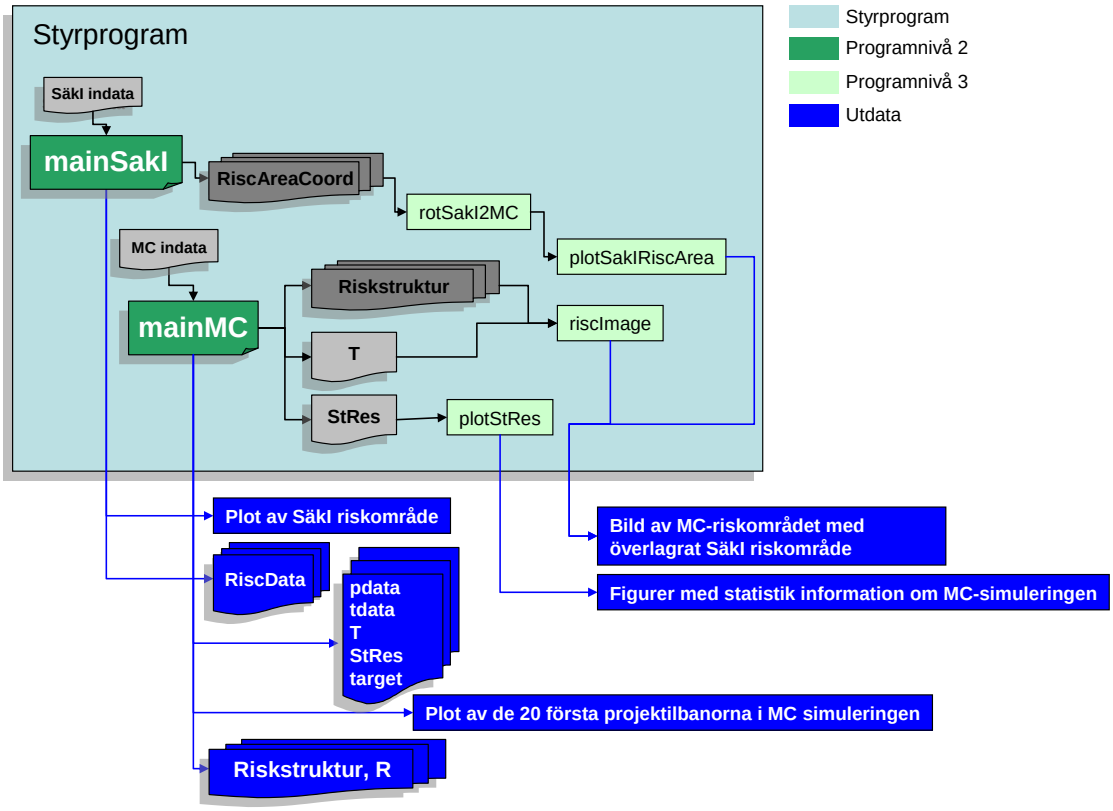
t1: Banstruktur för projektilbanan fram till studs, se bilaga 1.
t2: Banstruktur för projektilbanan efter studs, se bilaga 1.

Utdata:

tdata: Modifierad banstruktur enligt ovan.

2.3 Förening av SäkI- och Monte Carlo-riskområden

Under detta avsnitt förklaras hur de beräknade riskområden enligt avsnitt 2.1 samt 2.2 förenas till det önskade resultatet. Detta görs via ett enkelt styrprogram enligt schemat nedan.



Styrprogrammet ovan anropar funktionerna *mainSakI* (avsnitt 2.1.5.13) samt *mainMC* (avsnitt 2.2.5.1) för beräkning av SäkI:s riskområde respektive det Monte Carlo-simulerade riskområdet. Utdata ifrån dessa funktioner behandlas sedan med funktionerna under programnivå 3 i schemat ovan. Dessa funktioner finns beskrivna under avsnitt 2.3.1 samt 2.3.2.

Programnivå 2 i styrprogrammet ovan finns beskrivet under avsnitt 2.1.2 samt 2.2.3.

2.3.1 Funktioner för förening samt redovisning av riskområden

Tabell 4, MATLAB funktioner för förening av SäkI- och Monte Carlo-beräkningar

Namn	Beskrivning
MainXjobb_riscareas	Styrprogram
mainSakI	Huvudprogram för SäkI riskområden i MATLAB
mainMC	Huvudprogram för MC-simulerade riskområden i MATLAB
rotSakI2MC	Roterar koordinatmatriserna ifrån SäkI beräkningarna till att passa MC-beräkningarnas koordinatsystem
plotSakIRiscArea	Ritar upp SäkI riskområdet med hjälp av koordinatmatriser
plotStRes	Plottar statistiska resultatet lagrat i strukturen StRes
riscImage	Skapar riskbilden utifrån riskstrukturerna och topografin

2.3.2 Funktionsbeskrivningar

2.3.2.1 mainXjobb_riscareas

Styrprogram till skapandet av SäkI:s och de Monte Carlo-simulerade riskområdena. Programmet anropar funktionerna *mainSakI* samt *mainMC* för beräkning av riskområdena enligt avsnitt 2.2.1 respektive 2.2.2. Programmet redovisar resultatet ifrån dessa beräkningar med funktionerna listade nedan.

rotSakI2MC	(avsnitt 2.3.2.4)
plotSakIRiscArea	(avsnitt 2.3.2.5)
plotStRes	(avsnitt 2.3.2.6)
riscImage	(avsnitt 2.3.2.7)

Utdata ifrån ovanstående funktioner redovisas och samtliga strukturer i bilaga 1: *Struktursamling* åskådliggörs i MATLAB:s kommando fönster.

2.3.2.2 mainSakI

se avsnitt 2.1.5.13: *mainSakI*.

2.3.2.3 mainMC

se avsnitt 2.2.5.1: *mainMC*.

2.3.2.4 rotSakI2MC

Vid beräkning av SäkIs riskområden sker skjutning i y-led medan vid beräkning av Monte Carlo-simulerade riskområden sker avfyrningen i x-led. Denna skillnad gör att koordinatmatriserna i SäkI strukturen *RiscAreaCoord* behöver roteras för att anpassas till koordinataxlarna i MC-simuleringarna innan riskområdena kan överlagras. Rotationen som krävs motsvarar rotationsmatrisen ***R***:

2.3.2.3.e1

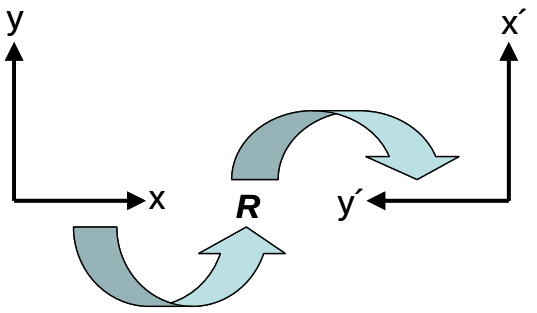
$$R = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$$

Effekten av denna rotationsmatris kan ses i figur 3.1 till höger.

Låt ***K*** vara en godtycklig koordinatmatris ifrån *RiscAreaCoord* enligt:

2.3.2.3.e2

$$K = \begin{bmatrix} x_1 & y_1 \\ \cdot & \cdot \\ \cdot & \cdot \\ \cdot & \cdot \\ x_n & y_n \end{bmatrix}$$



Figur 3.1, rotation av SäkI koordinataxlar, x och y, till MC-koordinataxlar, x´ och y´

Rotationen av matrisen ***K*** till matrisen ***K´*** blir:

2.3.2.3.e3

$$K'^{T} = R \cdot K^{T} = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \cdot \begin{bmatrix} x_1 & y_1 \\ \cdot & \cdot \\ \cdot & \cdot \\ \cdot & \cdot \\ x_n & y_n \end{bmatrix}^{T} = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \cdot \begin{bmatrix} x_1 & \cdot & \cdot & \cdot & x_n \\ y_1 & \cdot & \cdot & \cdot & y_n \end{bmatrix} =$$

$$= \begin{bmatrix} y_1 & \cdot & \cdot & \cdot & y_n \\ -x_1 & \cdot & \cdot & \cdot & -x_n \end{bmatrix} = \begin{bmatrix} x'_1 & \cdot & \cdot & \cdot & x'_n \\ y'_1 & \cdot & \cdot & \cdot & y'_n \end{bmatrix}$$

K:s y-koordinater har roterats till x´-koordinater och ***K***:s x-koordinater har blivit negativa y´-koordinater i enlighet med figur 3.1. Matrisen $\boldsymbol{K'^{T}}$ transponeras sedan tillbaka för att få matrisen ***K´***.

Ifall att avfyrning i MC-simuleringarna inte sker ifrån origo förflyttar rotSakI2MC även SäkI:s riskområden till att utgå från startpositionen för Monte Carlo simuleringen.

Syntax:

Res = rotSakI2MC(p, start)

Res = rotSakI2MC(p)

Indata:

p:	Struktur med koordinatmatriser för de olika områdena av SäkI:s riskområde, se bilaga 1, struktur *RiscAreaCoord*.
start:	1x3 matris som anger avfyrningspositionen i MC-simuleringen, *start* = [x_{start} y_{start} z_{start}]

Utdata:

Res:	Struktur med koordinatmatriser för de olika områdena av SäkI:s riskområde roterade till MC-simuleringarnas koordinataxlar

2.3.2.5 plotSakIRiscArea

plotSakIRiscArea skiljer sig ifrån *plotRiscArea* (se avsnitt 2.1.5.4) på så sätt att de matriser som endast är beräknade för halva SäkI:s riskområde speglas i linjen y = 0 istället för x = 0 för att anpassas till MC-simuleringarnas koordinataxlar.

Syntax:

plotSakIRiscArea(RiscAreaCoord)

Indata:

RiscAreaCoord: Struktur med koordinatmatriser för de olika områdena av SäkI:s riskområde efter rotationen med funktionen *rotSakI2MC*, se bilaga 1.

Utdata:

Figur: SäkI:s riskområde

2.3.2.6 plotStRes

Funktionen redovisar resultatet i strukturen *StRes*, se bilaga 1.

Syntax:

plotStRes(StRes)

Indata:

StRes: Struktur innehållande data om vad som sker under Monte Carlo-simuleringen, se bilaga 1.

Utdata:

Figur 1: Två histogram med avfyrningsvinklar

Histogram 1:	Elevationsvinklarna ifrån avfyrning.
Histogram 2:	Vinklarna i sida ifrån avfyrning.

Figur 2 Sex histogram med data över händelserna i studseffekten.

Histogram 1:	Nedslagsvinklar
Histogram 2:	Elevationsvinklar efter studs
Histogram 3:	Vinklar i sida före studs
Histogram 4:	Avvikelsen i sida efter studs
Histogram 5:	Avvikelsen i elevation efter studs
Histogram 6:	Förhållandet mellan infallshastigheten och utfallshastigheten

Figur 3: Pot av de 10000 första slutpositionerna i MC-simuleringen.

2.3.2.7 riscImage

Funktionen skapar riskbilden av det Monte Carlo-simulerade riskområdet med MATLAB-funktionen *image* [8]. *riscImage* tar in en riskmatris *R* [*non*x1], non = antalet noder i topografin, som innehåller riskvärdena för respektive nod (se teorin i avsnitt 2.2.1.3) och gör om den till en riskmatris R´ [*nox*x*noy*], *nox* = antalet noder i x-led, *noy* = analet noder i y-led (se avsnitt 2.2.5.4).

2.3.2.6.e1

$$R = \begin{bmatrix} r_1 \\ \cdot \\ \cdot \\ \cdot \\ r_{non} \end{bmatrix} \Rightarrow R' = \begin{bmatrix} r_{11} & \cdot & \cdot & \cdot & r_{1nox} \\ \cdot & \cdot & & & \cdot \\ \cdot & & \cdot & & \cdot \\ \cdot & & & \cdot & \cdot \\ r_{noy1} & \cdot & \cdot & \cdot & r_{noynox} \end{bmatrix}$$

Matrisen *R´* är skapad specifikt för användning i funktionen *image* [8] som tolkar värdena i matrisen som färger.

Med funktionen *riscImage* kan även upplösningen på bilden över riskområdet varieras med hänsyn till inparametern *refine*. Funktionen skapar extra noder mellan de topografiskanoderna på inbördes avstånd lika med parametern *refine* och interpolerar ut risken över dessa noder med MATLAB:s inbyggda funktion *interp2* [8].

Syntax:

riscImage(T, R, refine)

Indata:

T:	Topografistruktur, se bilaga 1.
R:	*n*x1 matris, *n* = antalet noder i topografin, med riskerna för de topografiska noderna.
refine:	Nodavstånd enligt ovan.

Utdata:

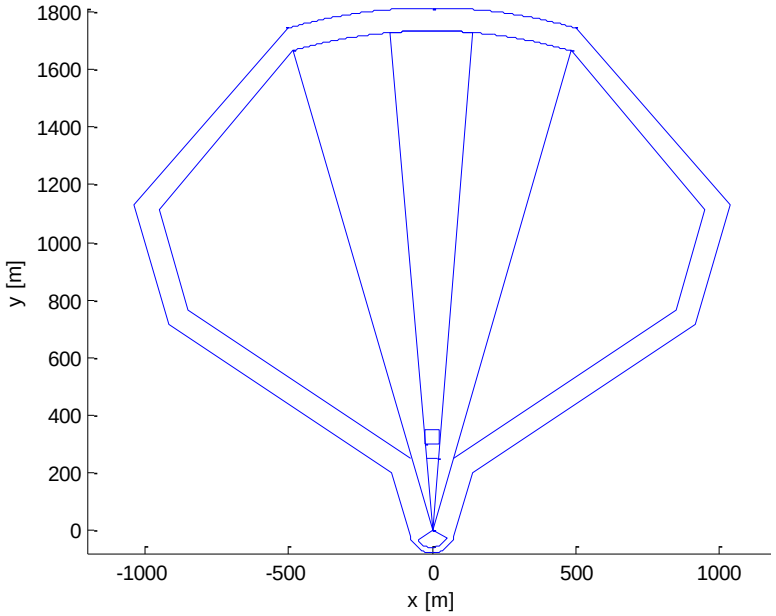
Figur:	Det Monte Carlo-simulerade riskområdet

3 Resultat

Resultatet av detta arbete är två MATLAB program för beräkning av SäkI:s riskområden (se avsnitt 2.1.2) samt beräkning av de nya Monte Carlo-simulerade riskområdena (se avsnitt 2.2.3). Till resultatet hör även program för sammankoppling av de båda riskområdena enligt avsnitt 2.3. Nedan följer exempel på resultat ifrån samtliga program.

3.1 SäkI

Resultat ifrån SäkI:s riskområdesberäkningar med fiktiva inparametrar redovisas nedan i figur R1.




```
SakIRiscData =

    Amax: 350
    Amin: 300
       M: 50
   alpha: 0.0831
       f: 50
       N: 0.5000
       V: 0.2000
      Nu: 1.0494
       k: 80
       Q: 0.7000
      k1: 80
       u: 60
       r: 0
    Dmax: 2600
      DC: 'other'
       c: 600
       l: 1385
       h: 1735
      cd: 0
      Q2: 0
treeline: []
```

Figur R1

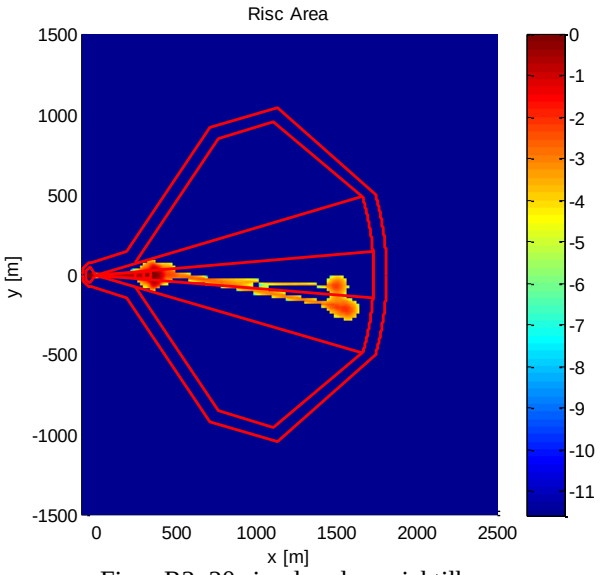
Parametrarna i strukturen *RiscData* (se bilaga 1) finns listade till höger om figuren i strukturen *SakIRiscData*.

Programmet för beräkning av SäkI:s riskområden ger det önskade resultatet. Samtliga områden definierade av strukturen *SakIRiscData* i teorin, avsnitt 2.1, åskådliggörs i det skapade riskområdet.

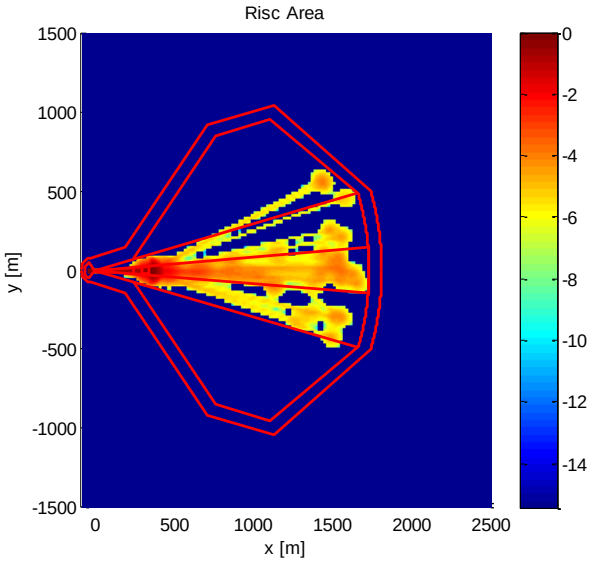
3.2 Monte Carlo-simulerade riskområden

Utifrån SäkI:s riskområde presenterat i avsnitt 3.1 redovisas här ett exempel ifrån programmet för Monte Carlo-simulerade riskområden.

Målet placerades på SäkI:s målområdes bortre gräns *Amax* (350m) och har en bredd av 70m och en höjd på 3m. Projektilens och projektilbanans parametrar, *pdata* respektive *tdata*, återfinns i bilaga 2.



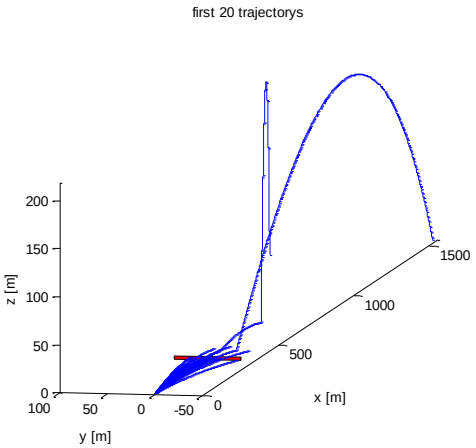
Figur R2, 20 simulerade projektilbanor



Figur R3, 200 simulerade projektilbanor

Avfyrningsvinklar har fördelats med normalfördelning i både sida och elevation enligt avsnitt 2.2.5.8. Studsberäkningsfunktionen som använts är *calcBounce_normPn* (avsnitt 2.2.5.16.2) och studsparametrarna, α och β, i funktionen *calcAlphaBeta* (avsnitt 2.2.5.24) är väl tilltagna för att områdena bättre ska passa till SäkI-beräkningarnas ”worst case” område.

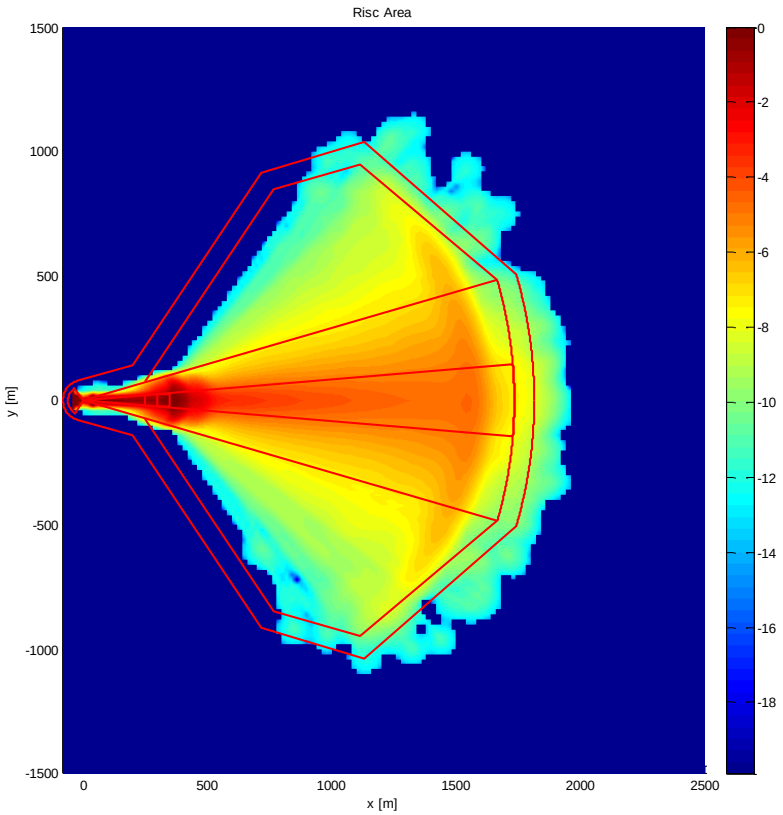
$\alpha = 0{,}8$
$\beta = 0{,}7$



Figur R4, de 20 första projektilbanorna

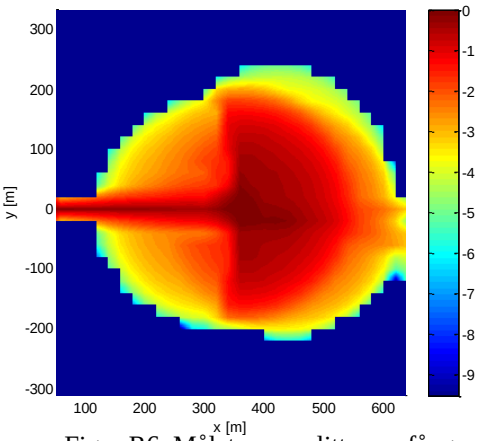
I figur R2 samt R3 åskådliggörs uppbyggnaden av det Monte Carlo-simulerade riskområdet, lägg märke till att riskerna redovisas i logaritmisk skala enligt färgkoden till höger om figurerna. Med så få simulerade projektilbanor förtydligas riskområdena ifrån passage samt splitter. Studseffekterna syns tydligt i figur R4 och även målet är möjligt att urskilja som en röd rektangel på $x = 350$m. Observera de stora studsarna. Dessa uppstår på grund utav de högt ansatta värdena på studsparametrarna α och β samt raketdriften. Det kantiga utseendet av det simulerade riskområdet beror på att *sz* = 20m (se avsnitt 2.2.1.3 samt struktur *T* i bilaga 2).

Till nästa figur har 200,000 projektilbanor simulerats och det sökta riskområdet börjar ta form. Områdena matchar varandra mycket väl med undantag från att det MC-simulerade riskområdet ser ut att vara något framskjuten i *x*-led i förhållande till SäkI området. Än en gång ska här belysas att det är fiktiva värden som använts i båda fallen. Om liknande resultat skulle fås med riktiga projektilparametrar skulle detta vara ett exempel på då man kan behöva definiera om riskområdet i längd eftersom att gula riskområden sträcker sig utanför SäkI:s riskområde. Detta skulle då även bero på vilken risk man ser som försumbar.



Figur R5, 200,000 simulerade projektilbanor

De största riskerna uppstår precis bakom målet på linjen *Amax* (se avsnitt 2.1.1.1, figur 1.3). Detta beror på att projektilen som simulerats har två laddningar. Den första laddningen fungerar som förpenetrator och detonerar efter tiden *at1* efter aktivering. Laddning 2 fortsätter igenom väggen och briserar efter tiden *at2*. Här åskådliggörs även inverkan av målet som splitteruppfång. Detta är dock inte övertydligt i denna figur på grund utav den låga upplösningen, stort *sz*, i förhållande till splitterparametrarna *k1* och *k2*. I figur R6 har detta förtydligats med en ny simulering av 20 projektiler där *k2*-värdet har ändrats till 200m. I denna figur kan man dra slutsatsen att minst en av projektilerna har briserat hitom målet och utifrån strukturen *StRes* för denna simulering ges parametern *mindist* av:



Figur R6, Målet som splitteruppfång

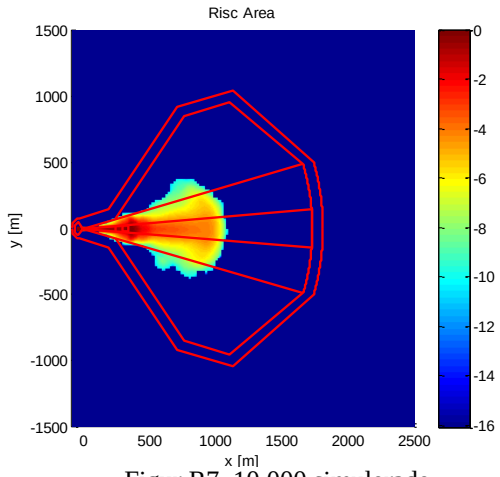
$$mindist = [304, -5, 0]$$

Alltså avslutades en projektilbana på avstånd x = 304m ifrån avfyrningsläget vilket är 46m kort om målet.

Inverkan av studsparametrarna illustreras med en ny simulering av 10,000 projektilbanor med studsparametrarna:

$$\alpha = 0{,}5$$
$$\beta = 0{,}2$$

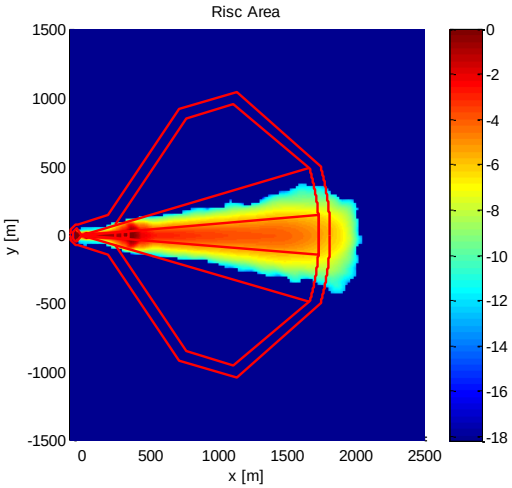
Denna simulering återfinns i figur R7.



Figur R7, 10,000 simulerade projektilbanor, α=0,5, β = 0,2

Inverkan av deviationen i markplan illustreras i figur R8. Här används samma fiktiva värden som i simuleringen med 200,000 projektiler med undantag att parametern *thethadev* i strukturen *T* har ändrats från 0,5 till 0,1.

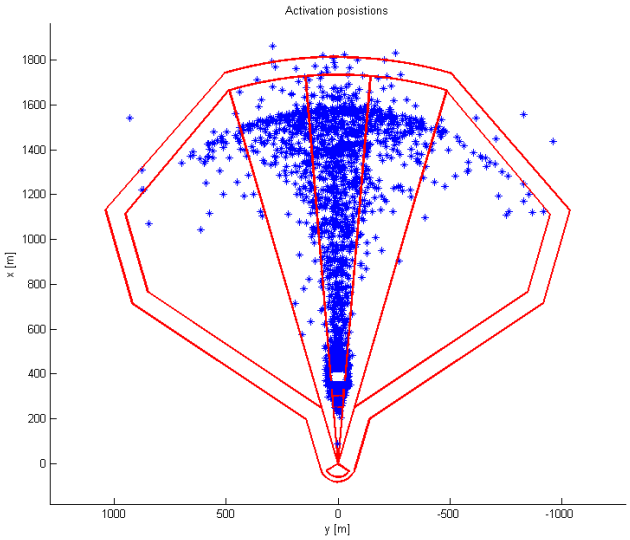
I simuleringen med 200,000 projektilerbanor fås aktiveringspositioner för projektilerna ifrån strukturen *StRes*. Dessa åskådliggörs i figur R9 samt R10. I figur R9, jämfört med figur R5 skulle man kunna anta att det ser säkert ut bakom målet. Detta är dock en felaktig uppfattning eftersom figur R9 endast anger aktiveringspositionerna för projektilerna. Briseringen sker sedan på avstånd motsvarande aktiveringstiderna *at1* respektive *at2*. Inga projektiler aktiveras i området bakom målet eftersom projektiler som når hit redan har aktiverats i väggen.



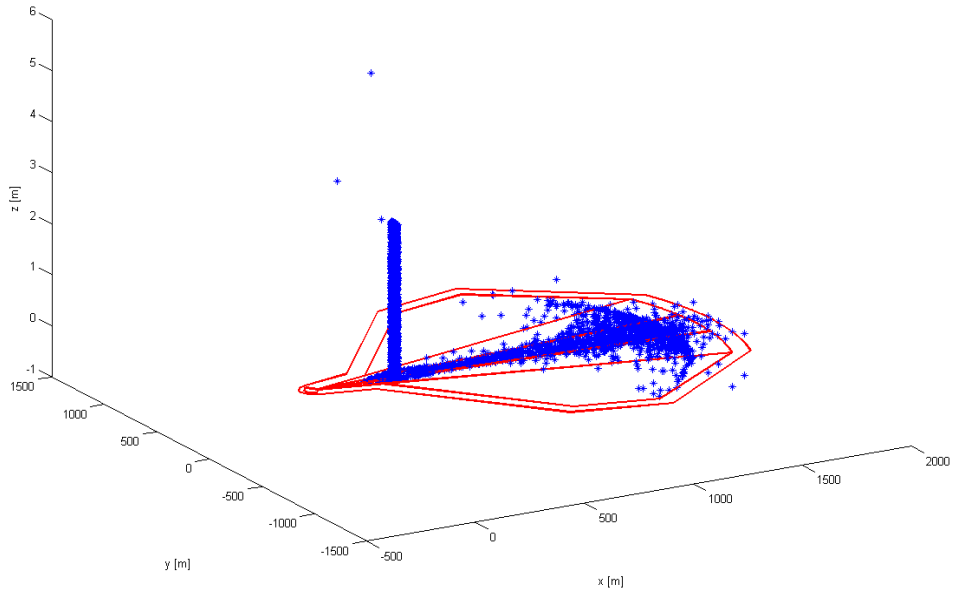
Figur R8, 100,000 simulerade projektilbanor, *thetadev=0,1*

Aktiveringspositionerna i målet illustreras tydligt i figur R10. Här åskådliggörs även tre briseringar som uppkommit på grund utav de binomialfördelade briseringssannolikheten ifrån konstruktionsfel (se flödesschemat: *Brisad i position Pb*, avsnitt 2.2.2).

I både figur R9 och R10 illustreras ”grounding” effekten, vilket är att en projektil slår i marken före målet. Detta motsvarar aktiveringspositionerna hitom målet ifrån avfyrningspositionen.



Figur R9, aktiveringspositioner för de 100,000 första projektilerna



Figur R10, aktiveringspositioner

Träffsannolikheten för simuleringen blev ca 60% enligt parametern *nrbOfHits* i strukturen *StRes*, bilaga 2.

I bilaga 2 återfinns även histogram över studsvinklar och avfyrningsvinklar ifrån simuleringen av 200,000 projektilbanor.

4 Diskussion och slutsatser

Överlag fungerar programmet för beräkning av riskområden som planerat. De simulerade områdena ger den definierade risken i avsnitt 2.2 i skjutområdet. I resultatet framgår tydligt vikten av studsparametrarna α och β. För pålitliga resultat behöver dessa därför estimeras med utomstående simuleringar. Riskområdena som bildas vid simulering kan användas som komplement till SäkI:s riskområden för diskussion om vad det är som ger dessa sitt utseende samt hur de kan reduceras.

SäkI

Programmet för beräkning av SäkI:s riskområden fungerar inte felfritt för godtyckliga parametrar. Detta beror på att programmet förutsätter att det finns geometriska lösningar till alla områden som definieras av parametrarna i strukturen *RiscData* samt teorin i avsnitt 2.1 och för godtyckliga värden kan det saknas fysikaliska förklaringar som där med kan sakna geometriska lösningar. Programmet uppfyller dock sitt syfte för existerande värden för vapensystem.

MC metoden

I metoden som används för Monte Carlo-simulering av riskområden finns det ett händelseförlopp som inte tas hänsyn till. Detta förlopp är om projektilen slår i marken före armeringstiden hos projektilen, studsar och sedan briserar på grund av konstruktionsfel efter armeringstiden. Detta skulle motsvara följande steg i flödesschemat *Brisad i position Pb* i avsnitt 2.2.2:

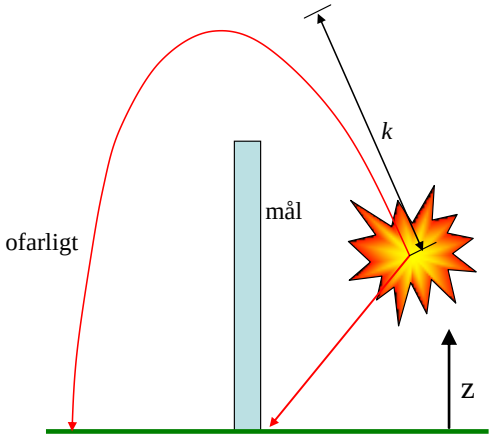
1) Beräkna bana från avfyrning till Pn
2) Ingen banbrisad före armering
3) Tiden *t_arm* har inte passerats
4) Målet träffas ej
5) Projektilen studsar utan att aktiveras
6) Ny bana från studs till nedslag
    - Armeringstiden passeras
    - Brisering på grund utav konstruktionsfel

Punkterna 1 till 6 ovan täcks av det nuvarande programmet, men briseringen som inträffar på grund utav konstruktionsfel efter studsen enligt punkterna efter punkt 6 kontrolleras inte. Programmet kontrollerar endast briseringar på grund utav konstruktionsfel före första studsen enligt flödesschemat. Teoretiskt blir risken för banbrisad efter armering mindre än den tänkta. De projektiler som slår ned i marken före armeringstiden tappar en av briseringsriskerna från konstruktionsfel. Risken att projektilen går i marken före armeringstiden är inte oväsentlig men beror helt på spridningen från avfyrningen. Risken att en projektil inte aktiveras i en studs är inte heller oväsentlig och beror på infallsvinkel.

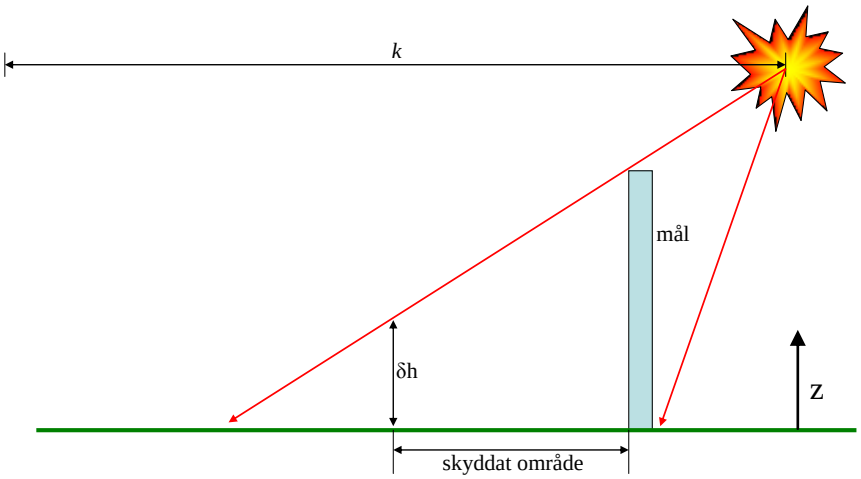
Sannolikheten för en ”förlorad” brisering blir produkten av sannolikheten att elevationsvinkel från avfyrning resulterar i ”grounding” före armeringsträckan, sannolikheten för banbrisad efter armering och sannolikheten att nedslagsvinkeln resulterar i studs utan aktivering. Risken för detta dubbelfel bedöms som försumbar och har därför inte tagits med i utförandet av programmet.

Målet som splitteruppfång

Sker en brisering på marken framför målet, eller på höjd under målets höjd, kan det ”regna” ner splitter på andra sidan målet, ungefär som om man kastar en näve grus över en vägg, se figur R10. Här antas det att splitter som på detta sätt far över målet inte utgör risk på andra sidan på grund utav dess låga energi. Så fort projektilen briserar över målets höjd returneras hela det cirkulära splitterområdet. Detta är dock inte fysikaliskt riktigt då målet fortfarande utgör ett skydd enligt figur R11 nedan.



Figur R10, ”splitterregn”



Figur R11, Brisad över målets höjd
k = splitter avståndet för projektilen
δh = höjd ovanför vilken splitter inte utgör risk

Dessa höjder borde undersökas på motsvarande sätt som de i figur 2.17 fast i tre dimensioner.

Man kan även argumentera att om brisaden sker tillräckligt nära kan splitter ha så pass mycket energi att de far rakt igenom väggen. Här behöver man då ansätta mer egenskaper till målet som till exempel: material och djup, samt lägga till beräkningar för splitterenergier. Dessa fall finns inte med i nuvarande programmet och målet som splitteruppfång kan endast ses som exempel på hur programmet kan användas och vidareutvecklas.

5 Vidareutveckling

Det finns en mängd punkter som kan utvecklas ytterligare i simulering av riskområden.

- Provning/simulering av studseffekter för bestämning av funktioner för beräkning av studsparametrarna α och β
- Utveckling av splitterområden
- Bestämning av topografiparametern *rhog* utifrån markunderlag
- Utveckling till sex frihetsgrader
    - Vindpåverkan
    - Ny studsberäkning
    - 3D topografi
        - Scenariosimuleringar

Vid vidareutveckling bör utveckling till sex frihetsgrader vara den första punkten som undersöks då denna påverkar hur resterande delar kan utvecklas.

En intressant utvecklingsmöjlighet är att använda programmet för scenariosimuleringar. Genom att skapa topografi utifrån verkliga fall där vapensystemet är tänkt att användas och utföra MC-simuleringar på dessa kan man då i utvecklingsstadiet av vapensystem testa och se vad som krävs för att få önskad verkan och samtidigt förutspå vilka risker som finns.

Passagerisk

Istället för att använda sig av punkterna i *passpoints* för framtagning av riskområdet för passage skulle avstånd till projektilbanan kunna beräknas med någon avståndsformel antingen i planet eller i rummet (se s 78, *”Linjär algebra”* [6]). Detta skulle resultera i mer exakta värden för passage i noderna.

Splitterrisk

Splitterområdena som uppstår vid brisad tar inte hänsyn till vilken typ av brisering som sket. Uppstår briseringen på grund utav stöt bör området påverkas av kollisionens egenskaper (så som hastighet, infallsvinkel med mera). De enda fall som skapar helt cirkulära splitterområden bör vara vid tidsmässig detonation på plant underlag eller kollision vid infallsvinkel 90^{o}.

Fördelning för splitterrisken avtar kvadratiskt från briseringspunkten ut till k-värdet för projektilen. Detta kan varieras enkelt i funktionen *calcFrgmRisc_k* om det är aktuellt att studera andra riskfördelningar. Risken behöver endast bero på avståndet från briseringspunkten, fördelningen kan sedan väljas godtyckligt.

Här finns även utrymme att lägga in andra villkor som kan orsaka brisering. Exempel på detta är:

- Träff av föremål i luftrummet
- Träff av föremål över markplan i topografi
    - Träd, buskar, m.m.

Tidsbestämd detonation

Vid tidsbestämddetonation kan den verkliga detonationstiden avvika från *dtimer*. Detta kan lösas genom att normalfördela *dtimer* i projektilstrukturen *pdata* (se bilaga 1) i början av varje loop i Monte Carlo-simuleringen med MATLABs slumpgenerator *normrnd* [8].

6 Antaganden

A.1: Luften är en ideal gas

A.2: Möjligt att estimera studsparametrarna α och β med utomstående simuleringar

A.3: Största risken vid studs uppkommer då projektilen endast utsätts för en riktningsändring.

A.4: Vid kollision antas projektilen vara orienterad så att nosen pekar åt hastighetens riktning

7 Referenser

[1] Försvarsmakten, *Säkerhetsinstruktion för vapen och ammunition med mera, Gemensam del, SäkI G 2008*, Sörman Information & Media AB

[2] Försvarsmakten, *Säkerhetsinstruktion för vapen och ammunition med mera. Skjutning med eldhandvapen, kulsprutor, pansarvärnsvapen samt lys- och signalammunition. Handgranater. Markeringsmedel. Säki Ehv/Pv 2008*, Sörman Information & Media AB

[3] Gunnar Blom, Jan Enger, Gunnar Englund, Jan Grandell, Lars Holst, *Sannolikhetsteori och statistikteori med tillämpningar*, Studentlitteratur 2005, femte upplagan

[4] Frank M. White, *Fluid Mechanics fifth edition*, Mc Graw Hill

[5] Niels Ottosen & Hans Petersson, *Introduction to the Finite Element Method*, Prentice Hall Europe 1992

[6] Gunnar Sparr, *Linjär Algebra*, Studentlitteratur 1982, 1994, andra upplagan

[7] Christer Nyberg, *Mekanik Grundkurs*, Liber AB

[8] The Mathworks inc, *MATLAB v. 7.6.0.324 (R2008a)*

8 Akronymer och förkortningar

Beteckning	Beskrivning
Ehv	Eldhandvapen
grg	Granatgevär
LTH	Lunds Tekniska Högskola
MC	Monte Carlo
nelm	Number of elements
non	Number of nodes
pskott	Pansarskott
Pv	Pansarvärnsvapen
SBD	SAAB Bofors Dynamics
slpsgr	Spårljuspansarspränggranat
SäkI	Försvarsmaktens säkerhetsinstruktioner

Bilaga 1

Struktursamling

Nedan finns samtliga strukturer för beräkning av SäkI- och Monte Carlo-simulerade riskområden

SäkI-strukturer

RiscData

Parameter	Beskrivning	Enhet
Amax	Största tillåtna skjutavstånd	m
Amin	Minsta tillåtna skjutavstånd	m
M	Bredd av skjutområdets främre gräns	m
f	Riskavstånd hitom skjutområdet	m
N	Mynningsvinkel	mrad
V	Riskvinkel för sidspridning	mrad
Nu	Riskvinkel i sida utanförskjutgränsens förlängning bakom vapnet	mrad
k	Riskavstånd för splitter	m
Q	Riskvinkel vid studs	mrad
k1	Riskavstånd för splitter, k1 linjen följer parallellt med V-linjen och rundas bakom vapnet	m
u	Riskavstånd bakom vapnet	m
r	Riskavstånd framför mynning	m
Dmax	Maximalt skjutavstånd	m
DC	Riskfall	text
c	Riskavstånd i sida vid studs	m
l	Riskavstånd bortom målområdet	m
h	Riskavstånd i längd	m
cd	Riskavstånd i sida vid studs av drivspegel	m
Q2	Riskvinkel vid skjutning mot skog	m rad
treeline	Avstånd till skog från utgångs läge	m
Alpha (α)	Vinkel åt höger (gräns höger) eller vänster (gräns vänster) från mittlinje	mrad

Samtliga parametrar i *RiscData* har definierats utifrån försvarsmaktens säkerhetsinstruktioner med undantag av *M* och *k1* (*SäkI G 2008*, kap 4, *Riskområden*, s. 49-68 [1] samt *SäkI Ehv/Pv 2008*, kap 8, *Riskområden*, s. 127-159, [2]).

RiscAreaCoord

Matris	Beskrivning
FA	Koordinater för skjutområdet
TA	Koordinater för målområdet
DngA_V	Koordinater för sidspridningen
DngA_f	Koordinater för markering av riskavståndet för direktträff hitom målområdet
DngA_Nu	Koordinater för riskområdet bakom vapnet
DngA_k	Koordinater för riskområdet ifrån splitter
DngA_rN	Koordinater för riskområdet framför mynningen till vapnet
DngA_Qc	Koordinater för riskområdet ifrån ricocheter

Monte Carlo-strukturer

R

Matris	Beskrivning	Värde
frgm	Matris av storlek nx1, n=non, anger splitterrisken	0-1
pass	Matris av storlek nx1, n=non, anger passagerisken	0-1
Nu	Matris av storlek nx1, n=non, anger risken bakomvapnet	0 eller 1
rN	Matris av storlek nx1, n=non, anger risken från mynningen	0 eller 1

StRes

Parameter	Beskrivning	Värde och/eller enhet
Eu0	Matris nx1, n = antal simuleringar. Elevationsvinklar som uppstått från avfyrning	rad
Su0	Matris nx1, n = antal simuleringar. Sidvinklar som uppstått från avfyrning	rad
E	Matris nx2, n = antal simuleringar. Nedslags vinkel samt elevationsvinkel före respektive efter studs.	rad
S	Matris nx2, n = antal simuleringar. Sidvinklar före sant efter studs	rad
V	Matris nx2, n = antal simuleringar. Fart hos projektil före respektive efter studs	m/s
endPos	Matris nx3, n = antal simuleringar. Slutpositionerna för de 10000 första projektilbanorna	m
mindist	Kortaste avstånd från avfyrningspositionen där ”grounding” uppstod.	m
maxH	Högsta höjden som nåtts i simuleringen	m
nbrOfhits	Antal projektiler som träffa målet	0-n, n = antal simuleringar
nbrOfBlind	Antal projektiler som ej briserat	0-n, n = antal simuleringar
nbrOfBnc	Antal projektiler som studsat	0-n, n = antal simuleringar

pdata

Parameter	Beskrivning	Värde och/eller enhet
nbrOfCharges	Antal laddningar	1-2
activationNbr	Parametervärde som anger när aktivering sker som kvoten mellan nedslagsvinkel (0-90) och kollisionsmaterialets hårdhet (1-5).	0-18
Hmin	Minsta höjd över vilken projektilen inte utgör risk. Kan ej vara mindre än största värdet av k1 och k2.	m
k1	Splitteravstånd för första eller enda laddningen. Finns två laddningar tillhör k1 förpenetratorn.	m
k2	Splitteravstånd för huvudladdningen om två laddningar.	m
Nu	Riskvinkel bakom vapnet.	rad
u	Riskavstånd bakom vapnet.	m
N	Riskvinkel framför mynning.	rad
r	Riskavstånd framför mynning.	m
befor_armingP	Sannolikheten att projektilen briserar före armering, binomialfördelat.	Väljs fritt
after_armingP	Sannolikheten att projektilen briserar efter armering, binomialfördelat.	Väljs fritt
armingT	Armeringstiden för projektilen.	s
at1	Aktiveringstiden för första laddningen.	s
at2	Akiverintstiden för andra laddningen.	s
dtimer	Timer för tidsbestämd detonation, sätts till 0 om projektilen inte har denna egenskap.	s
Dmax	Längsta möjliga skjutavstånd.	m
devFangles	1x3 matris som anger standardavvikelse eller intervall inom vilket avfyrningsvinklarna skall varieras.	Väljs fritt

T

Parameter	Beskrivning	Värde och/eller enhet
b	Topografins bredd modifierad så att ett helt antal element får plats på bredden.	m
l	Topografins längd modifierad så att ett helt antal element får plats på längden.	m
nelm	Antal element i topografin.	$l*2b/sz^2$
nen	Antalet noder som tillhör varje element.	4
noy	Antal noder i y-led.	2b/sz+1
nox	Antal noder i x-led.	l/sz+1
non	Totala antalet noder i topografin.	nox*noy
rhog	nx1 matris, n=nelm. Anger markens hårdhet samtliga element.	1-5, 1 = mjukt, 5 = hårt
coord	nx3 matris, n=non. Anger koordinaterna för varje nod, rad i ger koordinaterna för nod i.	m
Ex	nx4 matris, n=nelm. Anger x-koordinaterna hos noderna tillhörande ett visst element. Ex(i,j) anger x-koordinaten för nod j tillhörande element i.	m
Ey	Motsvarande Ex fast för elementens y-koordinater.	m
Ez	Motsvarande Ex fast för elementens z-koordinater.	m
Edof	nx5 matris, n=nelm. Anger vilka noder som tillhör vilket element samt elementets orientering. Första kolonnen i Edof anger element nummer därefter följer noderna som tillhör elementet.	Element och nodnummer

target

Parameter	Beskrivning	Värde och/eller enhet
b	Målets bredd i sidled vänster eller höger, total bredd=2b	m
h	Målets höjd	m
x	Avstånd i x-led till målet från avfyrningspositionen	m

tdata

Parameter	Beskrivning	Värde och/eller enhet
p_start	1x3 matris som anger avfyrningspositionen eller studsposition.	m
v_start	Begynnelsehastighet för projektilen ifrån avfyrningen eller efter studseffekten.	m/s
t_start	Starttid för beräkning av projektilbanan.	s
kCd	Luftmotståndskoefficient $a=-kCd*M^2$, M=machtalet	m/s^2
elevation	Anger riktningsvinkel i elevation från avfyrningspositionen eller elevationsvinkel efter studseffekten.	rad
sideangle	Anger riktningsvinkel i sida från avfyrning eller vinkel i sida efter studseffekten.	rad
rocket	Undertruktur som innehåller tider och accelerationer för bestämning av raketdriften.	Se struktur *rocket*
start_dir	1x3 vektor som anger begynnelseriktningen för projektilbanan	Normerad riktningsvektor
t	nx1 matris som anger tidpunkterna då differentialekvationen T2.e10 har lösts	s
p	nx3 matris som anger positionerna i de tidpunkter, t, då differentialekvationen T2.e10 har lösts	m
v	nx3 matris som anger hastigheterna i de tidpunkter, t, då differentialekvationen T2.e10 har lösts	m/s
absv	nx1 matris som anger farten i de tidpunkter, t, då differentialekvationen T2.e10 har lösts	m/s
t_top	Anger tidpunkten då projektilbanan var på sin högsta höjd	s
p_top	1x3 matris som anger projektilens position vid tiden t_top	m
v_top	1x3 vektor som anger projektilens hastighet vid tiden t_top	m/s

rocket

Parameter	Beskrivning	Värde och/eller enhet
t_start	Anger tidpunkt då raketdriften börjar.	s
t_raise	Anger hur lång tid det tar för raketdriften att nå full kraft.	s
t_stop	Anger tidpunkten då raketdriften avslutats.	s
t_fall	Anger hur lång tid det tar för raketdriften att avslutas.	s
a_start	Anger accelerationen vid starten av raketdriften.	m/s^2
a_stop	Anger accelerationen vid slutet av raketdriften.	m/s^2
kCd_reduction	Anger procentuell minskning av luftmotståndskoefficienten, kCd, under raketdriften.	0-1

Bilaga 2

Strukturparametrar ifrån beräkningarna presenterade i avsnitt 3: Resultat

```
pdata =

      nbrOfCharges: 2
     activationNbr: 6
              hmin: 300
                k1: 40
                k2: 80
                Nu: 0.8000
                 u: 60
                 N: 0.5000
                 r: 50
    before_armingP: 1.0000e-006
     after_armingP: 1.0000e-004
          armingT: 0.1600
              at1: 1.0000e-004
              at2: 0.0200
           dtimer: 0
             Dmax: 2200
        devFangles: [0.0050 0.0400 2]


    StRes =

              Eu0: [1x200000 double]
              Su0: [1x200000 double]
                E: [52625x2 double]
                S: [52625x2 double]
                V: [52625x2 double]
           endPos: [10000x3 double]
          mindist: [33.0785 1.0128 -2.1516]
             maxH: [927.2127 16.8420 -558.4406]
       nbrOfhits: 120378
      nbrOfBlind: 0
        nbrOfBnc: 37567
```

```
tdata =

      p_start: [0 0 -1]
      v_start: 180
      t_start: 0
          kCd:
    elevation: 0.0520
    sideangle: 0.0762
       rocket: [1x1 struct]
    start_dir: [0.9957 0.0760 -0.0520]
            t: [541x1 double]
            p: [541x3 double]
            v: [541x3 double]
         absv: [541x1 double]
        t_top: 1.0908
        p_top: [225.3351 17.2089 -6.2994]
        v_top: [224.2057 17.1226 5.6899e-016]

T =

            b: 1500
            l: 2580
            h: 0
           sz: 20
           ed: 0.2000
           sd: 0.2000
    thetadev: 0.5000
         nelm: 19350
          nen: 4
          nox: 130
          noy: 151
          non: 19630
         rhog: [19350x1 double]
        coord: [19630x3 double]
           Ey: [19350x4 double]
           Ex: [19350x4 double]
           Ez: [19350x4 double]
         Edof: [19350x5 double]
```

Histogram beräknade utifrån StRes