FFI RAPPORT

ERFARINGER MED AVAL - Simuleringsprogram for sårbarhet og våpenvirkning

HALSØR Marius

FFI/RAPPORT-2003/00480

FFIBM/798/139

Godkjent
Kjeller 29. september 2003

Stein Grinaker
Forskningssjef

ERFARINGER MED AVAL - Simuleringsprogram for sårbarhet og våpenvirkning

HALSØR Marius

FFI/RAPPORT-2003/00480

FORSVARETS FORSKNINGSINSTITUTT
Norwegian Defence Research Establishment
Postboks 25, 2027 Kjeller, Norge

FORSVARETS FORSKNINGSINSTITUTT (FFI)
Norwegian Defence Research Establishment

P O BOX 25
NO-2027 KJELLER, NORWAY

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (when data entered)

REPORT DOCUMENTATION PAGE

1) PUBL/REPORT NUMBER	2) SECURITY CLASSIFICATION	3) NUMBER OF PAGES
FFI/RAPPORT-2003/00480	UNCLASSIFIED	42
1a) PROJECT REFERENCE	**2a) DECLASSIFICATION/DOWNGRADING SCHEDULE**	
FFIBM/798/139	-	

4) TITLE

ERFARINGER MED AVAL - Simuleringsprogram for sårbarhet og våpenvirkning

EXPERIENCES WITH AVAL - Simulation program for Assessment of Vulnerability And Lethality

5) NAMES OF AUTHOR(S) IN FULL (surname first)

HALSØR Marius

6) DISTRIBUTION STATEMENT

Approved for public release. Distribution unlimited. (Offentlig tilgjengelig)

7) INDEXING TERMS IN ENGLISH:	IN NORWEGIAN:
a) Simulation program	a) Simuleringsprogram
b) Vulnerability	b) Sårbarhet
c) Lethality	c) Våpenvirkning
d) Target modelling	d) Målmodellering
e)	e)

THESAURUS REFERENCE:

8) ABSTRACT

AVAL is a simulation model for weapon effect and target vulnerability. Basically, it calculates the probability for a target to get a specific damage when fired upon by a specific weapon. Like every other simulation program, AVAL is dependent on good input data. The input data for AVAL is a weapon model and a target model. The target model describes the geometry of the target, specifies how much damage each component can sustain before it's rendered inoperative, and describes the consequences of a set of damaged components in the form of a fault tree. The weapon model describes the sensors, the penetration capacity of the warhead and the forming of behind armour debris.

Through an agreement with FMV, FFI has had the simulation program AVAL, version 4.2, for free trial until the summer 2003. In return, FFI shall provide a report on our experiences with AVAL. FFI have developed one target model and a number of weapon models, and conducted several simulations, primarily against targets that came with the model.

In this report we explain how we have operated the program, and what we consider to be its strengths and weaknesses. FFI has decided to acquire AVAL on a commercial basis after the expiration of our free trial period.

9) DATE	AUTHORIZED BY	POSITION
	This page only	
29. September 2003	Stein Grinaker	Director of Research

ISBN-82-464-0770-8

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (when data entered)

INNHOLD

ERFARINGER MED AVAL - Simuleringsprogram for sårbarhet og våpenvirkning

1 BAKGRUNN

FFI-prosjekt 798, Panserbekjempelse 2000+, har som hovedoppgave å finne et fremtidig konsept for panserbekjempelse. I tillegg ser vi spesielt på smart krum ild og aktive beskyttelsessystemer. Vi har også sett på anvendelsesområder for høyhastighetsmissiler, HVM. En egen aktivitet i prosjektet har vært å utvikle en motor for bruk i et slikt missil

For å løse oppgavene våre, foretar vi stridssimuleringer på taktisk og stridsteknisk nivå. Slike simuleringer er avhengige av gode inputdata for å gi meningsfulle resultater. Blant inputdataene vi har behov for, er sannsynligheten for at bestemte våpensystemer skal slå ut bestemte mål under gitte betingelser, også kalt P(kill), heretter betegnet P_k.

Den mest hensiktsmessige måten å skaffe data for P_k på, er å foreta sårbarhetssimuleringer. Tidligere har vi ved FFI benyttet programmet *Tankkill* (2) til slike simuleringer. Dette programmet begynner imidlertid å bli utdatert, og det ble besluttet at vi skulle se oss om etter alternative løsninger.

Prosjekt 798 er langt fra det eneste prosjektet ved FFI som har behov for å drive sårbarhetsanalyser. Alle som driver med stridssimuleringer har behov for data av typen P_k på en eller annen måte. For noen kan det være ønskelig å finne svake punkter ved våre enheter, slik at vi kan utbedre dette, og for andre kan det være interessant å vite hvor stor effekt, i form av for eksempel penetrasjon og splintdannelse, som skal til for å slå ut et mål med tilstrekkelig sannsynlighet. *Tankkill* kan kun benyttes for kjøretøy, men det kan ofte være behov for tilsvarende data for fly og fartøy.

I Sverige har FMV (Försvarets materielverk) og FOI (Totalförsvarets forskningsinstitutt) utviklet en sårbarhetsmodell de kalte *LIBRA*. Denne modellen så ut til å dekke de behovene vi hadde i prosjekt 798, og kunne i tillegg brukes til å beregne sårbarhet i andre mål enn kjøretøy. På grunn av navnerettigheter har programmet etter hvert endret navn til *AVAL*, som står for ”Assessment of Vulnerability And Lethality”. Det er FMV som eier rettighetene til programmet.

Etter å ha fått en presentasjon av programmet og gått på kurs i Sverige, inngikk vi en avtale med FMV om utlån av AVAL. Vi fikk låne programmet gratis ut juni 2003, mot at vi skulle skrive en rapport om våre erfaringer om bruken av AVAL.

Hovedformålet med denne rapporten er å beskrive våre erfaringer med bruken av AVAL, og kommentere sterke og svake sider. Der det er hensiktsmessig, kommer vi også med forslag til forbedringer. For å gjøre rapporten tilgjengelig også for lesere som ikke er kjent med programmet fra før, inkluderer vi litt informasjon om hvordan AVAL fungerer. Forfatteren

baserer denne informasjonen på erfaringer med bruk av programmet, samt informasjon som har fremkommet på diverse kurs.

2 GENERELT

AVAL er en simuleringsmodell for beregning av sårbarhet og våpenvirkning. I prosjekt 798 er vi primært opptatt av pansrede kjøretøy som mål, men AVAL kan også håndtere andre måltyper, som missiler, fly og fartøy. I forbindelse med studier av aktive beskyttelsessystemer (APS) har også missiler vært aktuelle som mål i prosjekt 798.

2.1 Oppgaver ved FFI

I perioden FFI har hatt AVAL til utlån, har vi blant annet ønsket å benytte modellen til følgende oppgaver:

- Bestemme effekten av Javelin og Spike skutt mot stridsvogn
- Bestemme effekten av en splintskur skutt fra et APS mot et missil
- Finne optimal størrelse på prefragmenterte splinter i MP-ammunisjon
- Se på betydningen av usikkerheten i tidsbrannrøret til MP-ammunisjon
- Studere sårbarheten og behovet for tilleggspansring for en M113
- Variere masse og hastighet til pilammunisjon og vurdere konsekvensene for P_k.

Disse oppgavene innebar generering av flere nye våpen, samt en M113 og enkelte mindre mål. De innebar også flere simuleringer, både sårbarhets- og effektsimuleringer, med diverse våpentyper mot flere mål.

Av forskjellige årsaker har vi ikke fått brukt så mye tid på arbeidet med AVAL som vi skulle ønske. Dels skyldes dette at programmet ble tilgjengelig senere enn antatt, dels at prosjekt 798 har vært underbemannet, og at arbeidet med AVAL derfor i perioder har måttet vike for andre oppgaver.

2.2 Oppbygging og virkemåte til AVAL

AVAL kan utføre tre typer simuleringer: Effektsimuleringer, sårbarhetssimuleringer og simuleringer av enkeltskudd (såkalt ”single warhead effect”). Den siste er mer for testformål enn for produksjon av reelle resultater.

AVAL er en stokastisk modell, og stokastiske modeller har behov for å generere tilfeldige tall. Randomgeneratoren i AVAL tar et tall som input (”seed”), og genererer ut fra dette en rekke tilsynelatende tilfeldige tall (pseudotilfeldige tall). Et bestemt seed gir en bestemt rekke med tall. I AVAL kan man selv bestemme hvilket seed som skal brukes. Det gjør det mulig å reprodusere en rekke med tilfeldige tall ved å bruke samme seed. Dette gir visse muligheter til å undersøke nøyaktig hva som har skjedd i en simulering. Hvis man synes noe er merkelig ved en bestemt simulering, kan man gjenta samme simulering med samme rekke med tall, og logge de resultater man er interessert i. Som regel er man imidlertid bare interessert i å generere en ny rekke med ”tilfeldige” tall.

AVAL simulerer på det nivået at man ser på effekten av ett enkelt våpen mot ett enkelt mål. I nåværende versjon tas ikke banen frem til målet med i simuleringen. Våpenets sensorer er derimot modellert. Når man simulerer enkeltskudd, simuleres ikke sensoren, kun stridshodet. I de andre simuleringsmodiene starter simuleringen med å bestemme når sensoren(e) trigger. Deretter settes stridshodet i våpenet av, eventuelt etter en tidsforsinkelse. Et våpen kan forøvrig inneholde flere stridshoder. På bakgrunn av tabeller som våpenet og målet er bygget opp av, beregnes så penetrasjon gjennom målet og dannelse av sekundærsplinter. For hver vitaldel som penetreres eller perforeres, trekkes det, basert på en gitt sannsynlighet, om komponenten blir slått ut eller ikke. Når denne biten av simuleringen er ferdig, testes det hvilke overordnede egenskaper ved målet som er ødelagt, basert på et feiltre og informasjonen om hvilke komponenter som er slått ut. Et feiltre er en logisk sammenheng mellom forskjellige komponenter og egenskaper til målet. Det beskriver hva som skal til for at målet skal være ute av stand til å utføre bestemte oppgaver, for eksempel bevege seg.

AVAL inneholder lite fysikk i seg selv. Enkelte ting, som spredning av brann og røyk, er innebygd, men brukeren har frihet til å justere de fleste effektene selv, basert på empiriske data eller egne modeller. Brukeren må altså selv beskrive våpenet gjennom dets egenskaper, som for eksempel hvor stor penetrasjonsevne det har og hvordan det danner sekundærsplinter, som funksjon av diverse parametre. For målet må hver enkelt komponent beskrives blant annet ved hva som skal til for å ødelegge komponenten og hvor mye energi et stridshode mister når det penetrerer komponenten. AVAL kan sees på som en avansert kalkulator, som beregner den totale virkningen et våpen har mot et mål, når brukeren selv beskriver hvordan målet er bygget opp, hvordan våpenet virker og hvordan diverse fenomener skal tas hensyn til.

3 GENERERE MÅLMODELLER

En målmodell i AVAL er bygget opp av vitaldeler og strukturdeler. Vitaldeler er komponenter som er av betydning for målets funksjon, mens strukturdeler kun har betydning som pansring. Komponenter kan bestå av kun ett materiale, eller de kan ha ett materiale som ”skall” og et annet materiale på innsiden. Materialegenskapene beskrives i en egen fil. For vitale deler angis også en sårbarhet, det vil si hvor mye komponenten tåler før den slutter å fungere. Et feiltre beskriver hvilke vitaldeler som må være slått ut for at målet skal ha mistet en bestemt funksjonalitet.

3.1 AVAL-CAD

AVAL-CAD er en AutoCAD-modul for innlegging av mål. Det er primært innlegging av målets geometri som forenkles ved bruk av AVAL-CAD, men AVAL-CAD inneholder også gode menyer for å gi de nødvendige egenskaper til målets komponenter. Menyene er stort sett enkle å manøvrere i, og inneholder den nødvendige funksjonalitet. Man må ha AutoCAD for å kunne bruke AVAL-CAD. Vi hadde ingen AutoCAD-lisenser ved FFI fra før, så vi måtte anskaffe en egen lisens for AVAL-formål. Selv om AVAL-CAD ikke forutsetter en inngående kjennskap til AutoCAD, valgte vi også å ta et grunnkurs i AutoCAD før vi startet å bruke programmet. Vi valgte å gå til anskaffelse av AutoCAD 2002. AVAL-CAD var tidligere kun testet med AutoCAD 2000 og R14. Det er små forskjeller på AutoCAD 2000 og AutoCAD 2002, og AVAL-CAD ser ut til å kjøre fint også under AutoCAD 2002. Riktignok har vi hatt problemer med funksjonen *lagre mål*. Dette *kan* selvsagt skyldes at vi bruker AutoCAD 2002, men antakeligvis skyldes det en feil i programmet. Vi kommer tilbake til dette i kapittel 3.2.

AVAL-CAD tar bare i bruk en liten del av mulighetene som ligger i AutoCAD. Blant annet finnes det i AutoCAD flere funksjoner for 3D-modellering som det ikke gjøres bruk av. Dette skyldes antakelig at AVAL krever et spesielt format på komponentene som målet består av: Komponentene skal være konvekse, og overflatene skal bestå av plane flater.

AVAL-CAD krever at man legger inn alle komponenter ”fra bunnen av”. Det er ikke mulig å ta en eksisterende AutoCAD-tegning og gjøre modifikasjoner på denne. Det hadde vært ønskelig at dette var mulig, da det er tungvint å måtte tegne på nytt en figur som allerede eksisterer i AutoCAD-format. Det ville vært en styrke om men kunne beholde geometrien i den opprinnelige figuren og legge til data som AVAL krever på denne figuren. Det er likevel en hjelp å ha en AutoCAD-tegning av målet tilgjengelig, slik at man kan ”tegne over” denne.

Det er viktig å huske på at enheten i AutoCAD normalt er mm, mens enheten i AVAL er meter. Dette førte til at målet vårt opprinnelig ble en faktor 1000 for stort. Dette var det overraskende vanskelig å korrigere. Det burde finnes en funksjon i AutoCAD for å skalere objekter, men vi fant ikke denne funksjonen og måtte gjøre korrigeringen manuelt i teksteditoren. Vi importerte tekstfila til Excel og gjorde endringene der, før vi importerte fila tilbake til tekstformat. Dette tok en god del tid.

Det ville vært fint om man kunne benytte flere av funksjonene i AutoCAD ved innlegging av mål, eller aller helst bruke et ”ekte” 3D-modelleringsverktøy, som for eksempel Inventor eller

SolidWorks. Men antakelig ville dette innebære alt for omfattende endringer av AVAL til at det lønner seg. Dessuten er det ikke innleggingen av målets geometri som er den tyngste biten – det er jobben med å skaffe sårbarhetsdata som er dimensjonerende.

Første punkt for oss var å skaffe en tegning av en M113. På fellesverkstedet ved FFI fantes det en full 3D-tegning av M113, men ikke i AutoCAD (AutoCAD er egentlig ikke et 3D-verktøy). Det var imidlertid mulig å overføre denne tegningen til et format som AutoCAD klarte å lese. Denne tegningen var ikke komplett, og det var flere komponenter som var utelatt. Dette er imidlertid et datainnsamlingsproblem, og ikke et AVAL-problem.

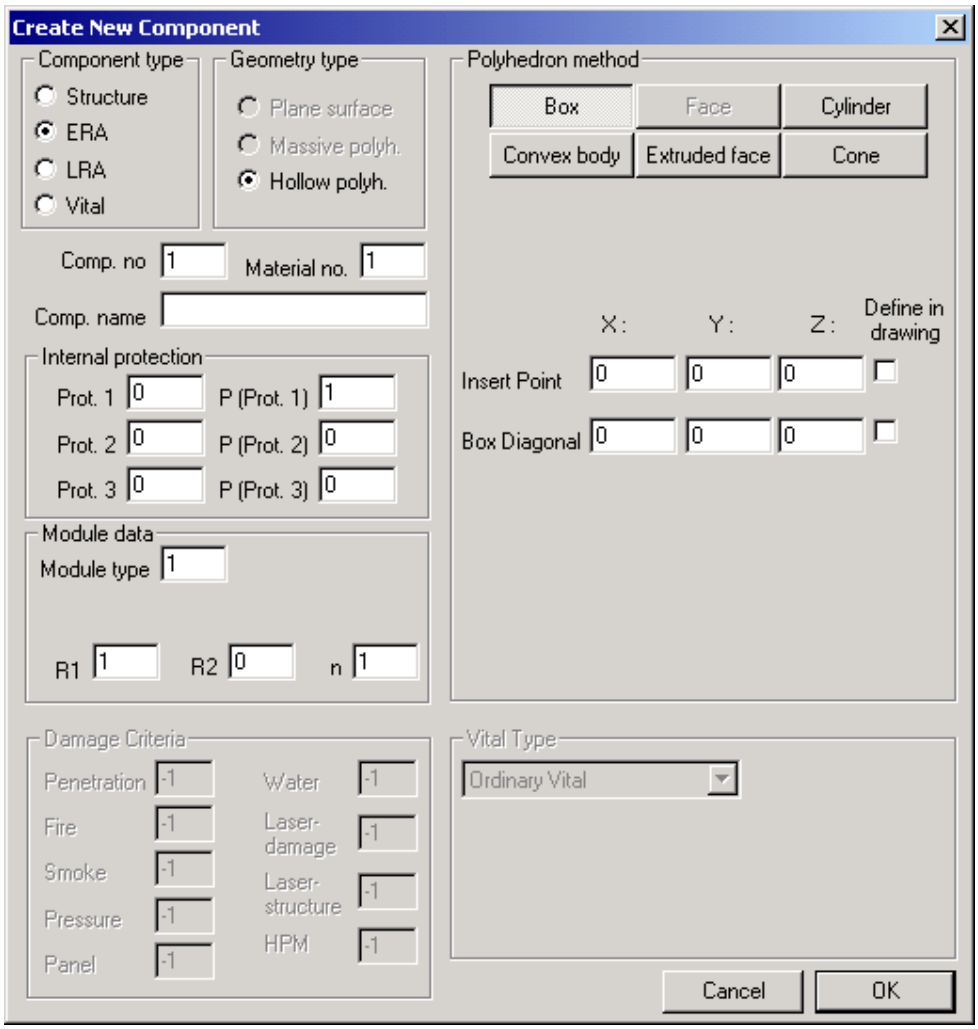
Vi gikk så i gang med å lage komponenter i AVAL-CAD. Dette ble gjort ved å forsøke å ”tegne over” tegningen vi hadde fått fra FFIs fellesverksted. Når vi tegner i 3D i AutoCAD, må vi enten oppgi koordinater eksakt, eller bruke et eksisterende punkt i figuren som vi ”låser” til. Dersom man prøver å lage et punkt ”like ved” et eksisterende punkt, vil AutoCAD måtte legge dette i z=0-planet, da den ikke kan vite hvor langt ”inn i skjermen” vi forsøkte å plassere punktet. Dette er ganske opplagt. For de fleste komponentene gikk det greit å ”tegne over”, men det oppsto noen problemer med de komponentene i modellen vi hadde fra før som var laget med 3D-funksjonaliteten i AutoCAD, som AVAL-CAD ikke kan benytte. Slike figurer har ikke nødvendigvis så mange punkter å ”låse” på – de kan være lagret som en sylinder med en bestemt radius. Da er det kun to senterpunkter å låse til – ingen punkter langs randen. For en hel sylinder har AVAL-CAD egen funksjonalitet som håndterer dette, men for en kvart sylinder, for eksempel, må man lage punkter langs randen, og her er det altså ingen punkter å låse til. En slik sylinder lages i AVAL-CAD som en ”extruded face”, det vil si at man lager en flate ved hjelp av punkter, og genererer så en sylinder med dette som base. Man trenger da punkter å låse flaten til. Vi løste dette problemet ved å opprette ”hjelpefigurer” som hadde de punktene vi trengte å låse til.

Komponenter i AVAL kan ha seks typer geometri: Box, Face, Cylinder, Convex body, Extruded face og Cone. Med unntak av ”Face” kan alle disse lages ved hjelp av ”Convex body”, så egentlig er det bare to typer. Komponenter som ikke er av typen ”face”, kan være enten hule eller massive. Hule komponenter er mest fleksible, og det er som oftest disse man ønsker å bruke for å beskrive en komponent, med mindre komponenten kun består av ett materiale og har en enkel geometri. Vi brukte massive komponenter i belteplatene, men de fleste andre komponentene ble modellert som hule.

Flater i AVAL (”face”) har den merkelige egenskapen at de har en ”utside” og en ”innside”, så det er viktig å passe på hvilken vei man plasserer den på målet. Vår erfaring er at i de aller fleste tilfeller er det greiere å lage en tynn, massiv boks enn å benytte flater.

Hule komponenter i AVAL kan beskrives i detalj via data for ”internal protection”, se Figur 3.1. Her angir man en sannsynlighetsfordeling for tettheten av materiale som en våpeneffekt vil møte når den penetrerer komponenten. Man angir tre sannsynligheter og tre tilhørende tettheter. For en stokastisk modell som AVAL, er dette en god måte å modellere komponenter på fremfor å gi dem en ”gjennomsnittlig” tetthet. Komponenter med slik ”gjennomsnittlig” tetthet vil ha den egenskapen at noen våpeneffekter alltid akkurat ikke klarer å perforere dem, mens andre alltid akkurat så vidt klarer det. Det er ofte mer realistisk at våpeneffekten kan perforere komponenten ved treff i ett område, mens den ikke klarer det i andre områder. Denne funksjonaliteten er

ypperlig for litt store komponenter som man ikke ønsker å beskrive i detalj (for eksempel en motor).



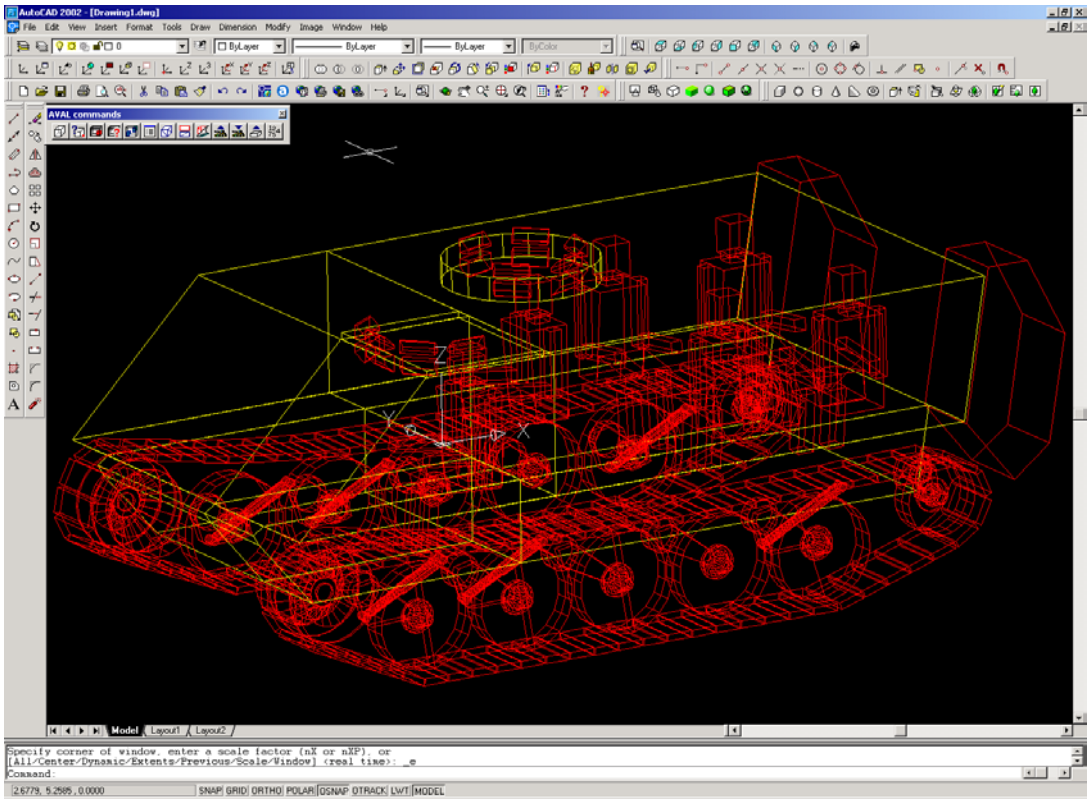
Figur 3.1"Create new component"-vinduet i AVAL-CAD

På grunn av måten AVAL håndterer penetrasjon på, kan man kun modellere konvekse komponenter i AVAL-CAD. Selvsagt er ikke alle reelle komponenter konvekse. Imidlertid kan man modellere andre komponenter som flere konvekse komponenter. Dette var litt klønete de første gangene vi prøvde, men etter hvert ble det uproblematisk. Man må bare huske på at man har modellert én komponent som flere komponenter når man senere skal lage feiltreet.

Etter hvert som vi begynte å få på plass en del komponenter, ønsket vi å gjøre modifikasjoner på noen av de komponentene vi allerede hadde laget. For enkeltkomponenter var dette lett å få til – menyen for å editere egenskaper i AVAL-CAD er god. Imidlertid savner vi en funksjon for å legge bestemte egenskaper på mange komponenter samtidig. Vi ønsket å endre materialet i belteplatene, og den eneste måten vi klarte å få til dette på, var ved manuelt å endre materialnummeret for hver enkelt belteplate. Heldigvis er det litt hjelp i at alt lagres som tekstfiler, men vi savner likevel en slik funksjon.

Måten vi endte opp med å gjøre dette på, var å gå inn i tekstfila som beskrev belteplatene. Der søkte vi på hele linja som inneholdt materialnummeret, og erstattet linja med en lik linje der

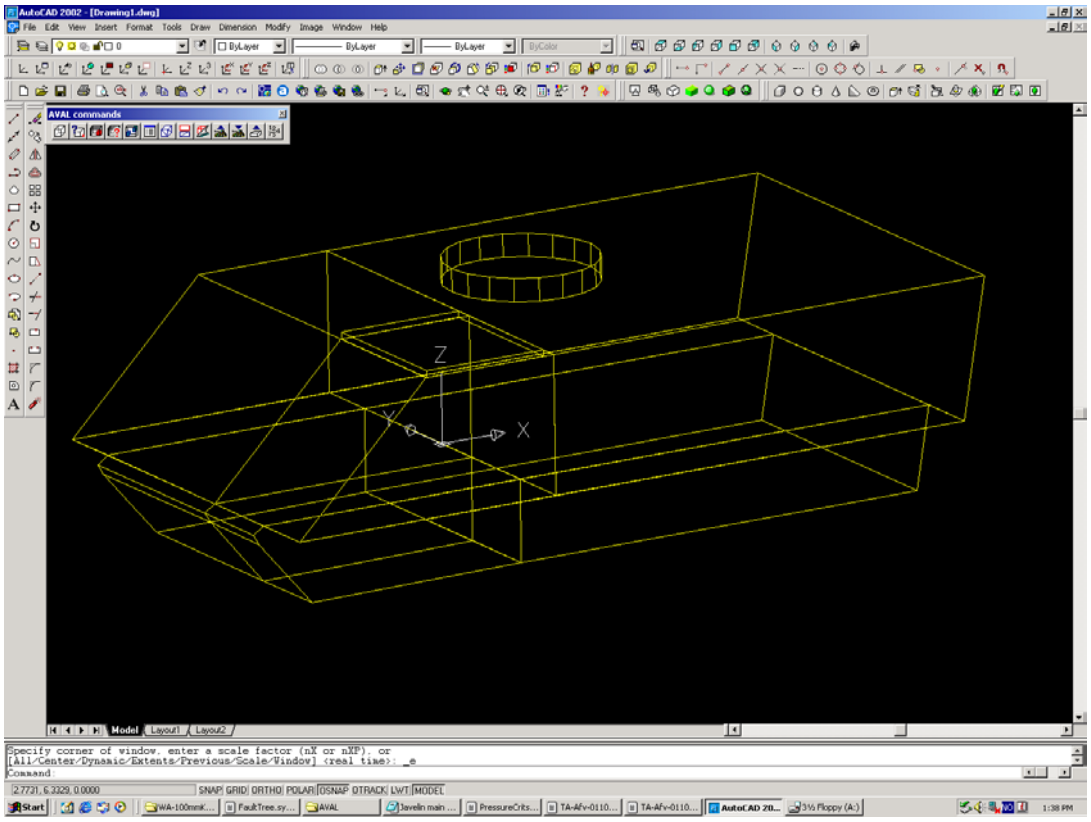
materialnummeret var byttet ut. Heldigvis hadde vi ikke flere komponenter i samme fil, så vi kunne erstatte alle linjene samtidig.



Figur 3.2 Modell av M113 i AVAL-CAD.

Når vi lagrer komponenter i AVAL-CAD, lagrer vi forskjellige grupper av komponenter i forskjellige filer: Beltene i én fil, personell i en annen fil, drivverk i en tredje fil og så videre. AutoCAD har funksjonalitet for å skille mellom forskjellige ”lag”. Når man laster inn et mål, havner komponenter i samme fil i samme lag. Disse lagene kan så skrus av og på alt etter hva man er interessert i å editere. Dette er en stor fordel, da det kan være vanskelig å jobbe med alle komponentene oppe på skjermen samtidig. Vi skulle imidlertid ønske at AVAL-CAD benyttet seg av denne funksjonaliteten i enda større grad. Slik det er i dag, må man alltid endre egenskaper for én og én komponent. Vi ser for oss en meny for å modifisere alle komponentene i et lag samtidig. Dette fordrer at det bare er én type komponent (vitaldel, strukturdel, ERA, etc.) i samme lag, men det er allerede slik i dag at det kun er mulig å lagre komponenter av samme type i samme fil, så det burde ikke være noe problem.

En av fordelene med å kunne skru av og på lag er illustrert i Figur 3.2 og Figur 3.3. Med alle komponentene i skjermbildet kan det bli ganske kaotisk. Det er ikke lett å markere akkurat de komponentene man er ute etter. Enda vanskeligere blir det å feste noe til akkurat riktig punkt, da det er veldig mange punkter tett opptil hverandre i Figur 3.2. I Figur 3.3 er alle lagene unntatt ”skrog” skrudd av. Her er det mye enklere å jobbe.



Figur 3.3 Modellen med alle lagene unntatt "skrog" skrudd av.

Da vi hadde modellert alle komponentene vi hadde på tegningen vår (i alle fall geometrien deres), ønsket vi å generere volumer.

3.2 Volumer

AVAL krever en egen fil for volumdata dersom andre effekter enn kun penetrasjon skal tas hensyn til. Både trykk-, brann- og røykeffekter avhenger av data i volumfila. Volumer i AVAL benyttes for å holde orden på hvilke komponenter som hører sammen i ett volum, hvilke volumer de forskjellige vitale komponentene befinner seg i, om det er noen varme overflater i et volum, om det er noen vitaldeler festet på utsiden av volumet og lignende.

Det er ikke trivielt å få på plass alle data om volumer. Menyene i AVAL-CAD virket også på dette området å være bedre enn de i selve AVAL-vinduet. Imidlertid kunne vi ønske å få opp en meny med en liste over alle eksisterende volumer når vi velger ”editer volum”, ikke bare muligheten til å angi kommandoer på kommandolinja. Dette er bare en vanesak, og etter hvert som vi har brukt programmet en del, er det ikke lenger noe problem. I begynnelsen forventet vi imidlertid å få opp et vindu, fordi det gjorde vi ved ”lage volum”. Vi føler derfor at et vindu vil være mer intuitivt enn å benytte kommandolinja.

Et annet problem gjelder lagring. I AVAL-CAD finnes det to muligheter for lagring: Lagre komponent og lagre mål. Lagre komponent lagrer informasjon om utvalgte komponenter i en egen fil. Lagre mål skal antakeligvis lagre all informasjon om målet, også når det gjelder volumer. Denne funksjonaliteten fikk vi imidlertid aldri til å fungere. Når vi forsøkte dette, fikk vi feilmeldingen ”unhandled exception (access violation reading 0x0152)”. (Dette er for øvrig

det eneste vi ikke har fått til å fungere i AVAL så langt.) Det var av den grunn ikke mulig å lagre informasjon om volumer via AVAL-CAD. Vi kopierte derfor volumfila fra et eksisterende mål og editerte denne ved hjelp av en teksteditor. Det er en klar styrke at AVAL lagrer filer på tekstformat, slik at de er lette å lese og editere manuelt. Vi kommer tilbake til flere fordeler med dette senere. Selv om problemet lot seg løse på denne måten, hadde det selvsagt vært bedre om vi hadde kunnet gjøre dette i AVAL-CAD. Vi antar at dette problemet vil bli rettet opp i neste versjon, med mindre problemet skyldes kompatibilitetsproblemer med AutoCAD 2002. Dette er i så fall det eneste problemet med AutoCAD 2002 som vi har kommet over.

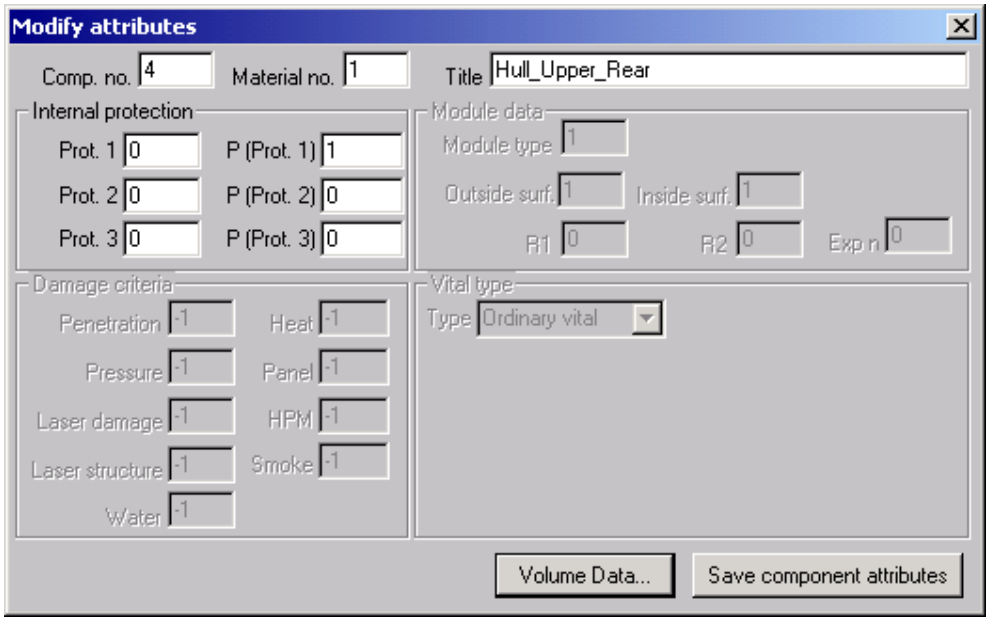
Det er naturlig nok bare hule komponenter som kan være med og danne et volum. Faktisk er det bare hule strukturdeler, vitaldeler får ikke danne volumer. Vi har så langt ikke følt noe behov for å la en vitaldel være del av et volum, men vi ser ikke hvorfor det ikke skulle være mulig dersom man ønsker det.

Siden hule strukturdeler kan være del av et volum, har disse bestemte egenskaper som er av betydning for volumet, kalt volumdata. Dette er data som er knyttet til hver enkelt komponent, og ikke til selve volumet som sådan. Det er egne data for hver flate i komponenten, og dataene beskriver arealet av flaten, eventuelle åpninger i flaten, hvorvidt flaten er gulv, vegg eller tak, hvorvidt flaten er sensitiv overfor trykk og/eller vannlekkasje, samt hvilke andre komponenter flaten ligger inntil. Menyen er fin, og dette er stort sett data som er enkle å legge inn, men vi kunne ønske oss at programmet selv beregnet arealet til overflaten. Det burde være trivielt, ettersom alle punktene på randen til flaten er kjent. Det tar faktisk litt tid å beregne disse arealene for hånd, selv om denne tiden selvsagt er neglisjerbar sammenlignet med den totale tiden det tar å generere et mål.

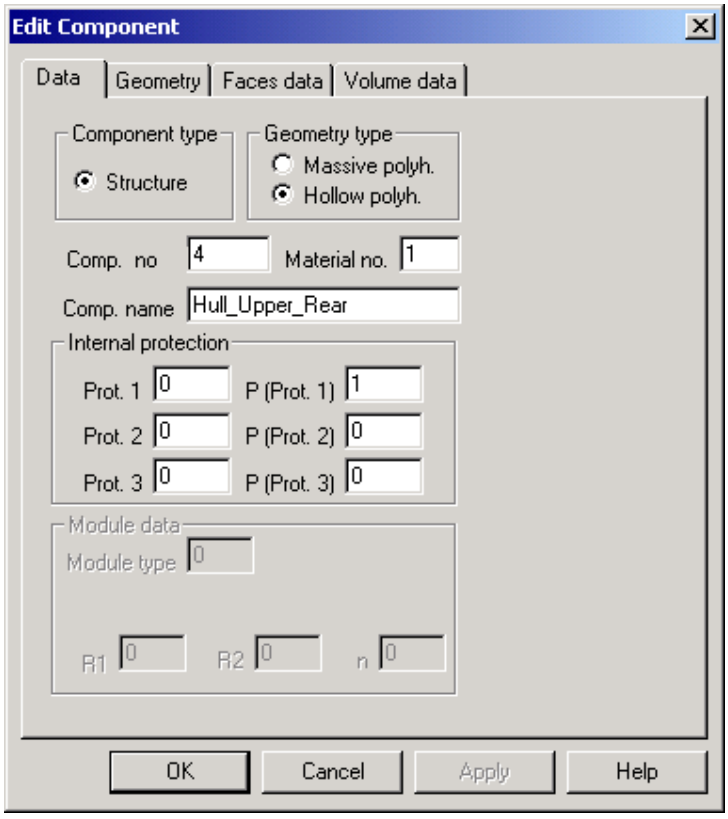
3.3 Funksjonalitet i AVAL-vinduet

I tillegg til AVAL-CAD, finnes det også i selve AVAL-vinduet funksjonalitet for å generere og modifisere mål. Menyene her er imidlertid ikke så gode som i AVAL-CAD. Vi finner det vesentlig enklere å generere og modifisere mål i AVAL-CAD enn i AVAL-vinduet. Dette gjelder ikke bare for geometri, men alt som har med målet å gjøre.

Det ville vært enklere å bruke funksjonaliteten i AVAL-vinduet dersom menyene var de samme som under AVAL-CAD. Programmet blir vanskelig å orientere seg i når utseendet på vinduene og innholdet i menyene ikke er like. La oss se på modifisering av enkeltkomponenter som et eksempel.



Figur 3.4 "Modify attributes"-vinduet i AVAL.

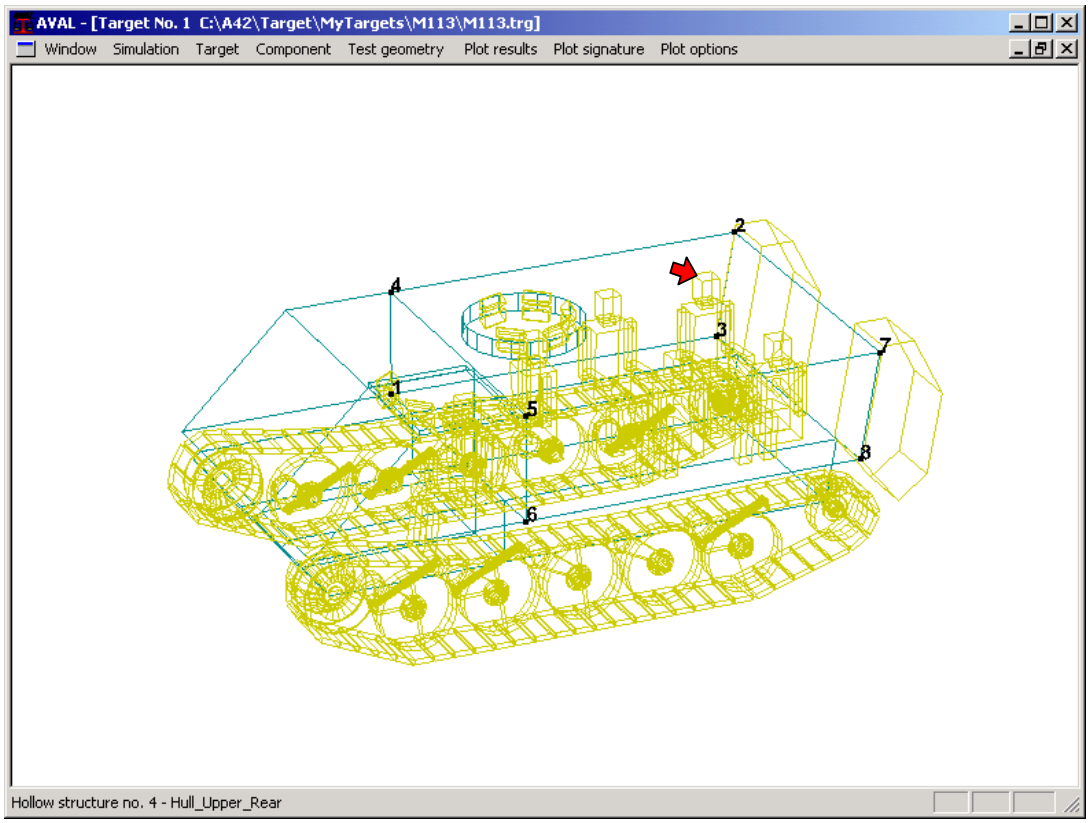


Figur 3.5 "Edit Component"-vinduet i AVAL-CAD.

Det hadde gjort det enklere om disse menyene var mer like. Slik det er nå, er det vanskelig å finne frem til den menyen man ønsker for å endre en bestemt parameter. Ikke bare har vinduene forskjellig utseende, de inneholder også forskjellig funksjonalitet.

Hvis man skal se på eller modifisere egenskapene til en komponent, er det greieste å bruke funksjonaliteten ”enable mouse selection”, som gjør det mulig å velge komponenter ved å klikke på dem med musa. Ulempen med denne metoden er at selv om man har en trådmodell på

skjermen, er det dessverre ikke mulig å velge indre komponenter på denne måten. Dette betyr at selv om man klikker der den røde pila markerer i Figur 3.6, er det ikke hodet på soldaten, men skroget på kjøretøyet som blir valgt. Det hadde vært å foretrekke om man kunne velge komponenter på samme måte som i AutoCAD (der ville hodet blitt valgt).



Figur 3.6 Trådmodell av et mål i AVAL-vinduet.

Dette problemet er det imidlertid mulig å jobbe seg rundt ved å ”skru av” strukturdelene. Det er også mulig å velge komponenter fra en liste, men da må man huske nøyaktig hva komponenten heter. Hvis det finnes flere komponenter med samme navn (ikke uvanlig), må man også huske nummeret på komponenten, noe som ikke er enkelt med tanke på at man har mange hundre komponenter og i tillegg av og til har behov for å renummerere komponentene i AVAL-CAD.

3.4 Nødvendige data

Det store problemet med innlegging av et mål har vært tilgangen til nødvendige data. Foruten geometrien til hver enkelt komponent som inngår i målet, trengs blant annet følgende informasjon:

- Materialegenskaper til alle materialene som inngår i målet
- Sårbarheten til hver enkelt komponent
- Feiltre
- Spesielle parametre som beskriver utvikling av røyk, brann, lekkasje og så videre.

Det er en del jobb å få dette på plass, men det må sees på som en fordel at brukeren har kontroll med disse parametrene. Alternativet ville vært at dette på en eller annen måte lå hardkodet og skjult i modellen, hvilket ville vært lite ønskelig. Alle disse dataene er viktige, men dataene som

omhandler sårbarheten til de enkelte komponentene er kanskje spesielt viktige. Disse dataene er dessverre også vanskelig tilgjengelige, og den klart vanskeligste jobben er i våre øyne å skaffe slike data.

Som for alle simuleringsprogrammer gjelder også for AVAL at uten gode inputdata får man heller ikke gode resultater. Man må derfor forsikre seg om at man har gode data for sårbarheten til alle komponenter. Hvordan man kan skaffe slike data, er et interessant spørsmål. Man kan selvsagt utføre eksperimenter, men dette er både kostbart og tidkrevende. For enkelte komponenter finnes det kanskje erfaringsdata (for eksempel for sårbarheten av personell). For noen veldig sensitive komponenter kan man kanskje sette sårbarheten til 1 uansett penetrasjonsdybde. Generelt er det imidlertid vanskelig å bestemme sårbarheten til en komponent. Dette vil være et område der brukerne av AVAL vil ha stort utbytte av å samarbeide og utveksle erfaringer og informasjon.

Feiltreet legges inn som en tekstfil. Det samme gjelder en god del annen informasjon i AVAL, som for eksempel materialdata. Når man er inne i programmet og målmodellen, er det greit å lese slike filer, men spesielt feiltreet kan det være en fordel å dokumentere separat, slik at det også kan være forståelig for andre enn de som har laget målmodellen.

3.5 Oppsummering

- **Funksjonen ”lagre mål” i AVAL-CAD virker ikke.** Dette er antakeligvis kun en feil i det programmet vi har hos oss nå, og det forventes å være i orden i den neste versjonen vi mottar. Funksjonen ”lagre komponenter” fungerer bra.
- **Geometrien til mål kan ikke importeres fra AutoCAD.** Selv om man har en AutoCAD-modell av målet, må man tegne hver enkelt komponent på nytt. Det ville vært enklere om man kunne ta utgangspunkt i en eksisterende modell og legge til de AVAL-spesifikke egenskapene på komponentene i denne modellen.
- **3D-mulighetene i AutoCAD tas ikke i bruk.** AVAL-CAD gir ikke rom for å bruke flere av de mulighetene som ligger i AutoCAD når det gjelder modellering i 3D. Vi tror dette skyldes måten AVAL utfører beregninger på, og antar at dette er noe man må regne med vil forbli slik i kommende versjoner. Innlegging av geometri er uansett ikke det mest tidkrevende i AVAL.
- **Flate komponenter har en ”innside” og en ”utside”.** Dette gjør at man må være svært påpasselig med hvordan man plasserer en flat komponent. Det er imidlertid stort sett bedre å modellere slike komponenter som tynne, massive bokser.
- **Man kan kun endre egenskaper til én komponent om gangen.** Av og til ønsker man å endre egenskapene til alle komponentene i et ”lag”. Det ville vært tidsbesparende om det fantes en funksjon for å legge ikke-geometriske egenskaper på flere komponenter samtidig.
- **Forskjellige menyer i AVAL og AVAL-CAD.** Menyene i AVAL-CAD er gode og enkle å bruke. Med de unntak som her er nevnt, er i det hele tatt AVAL-CAD et godt program. I AVAL er det lagt opp til at man skal kunne gjøre noen av de samme endringene som i AVAL-CAD, men menyene i AVAL ser ikke ut som menyene i AVAL-CAD, og har heller ikke samme funksjonalitet. Det ville vært en fordel om disse menyene var mer like.
- **Vanskelig å velge ”interne komponenter” i AVAL.** Når man prøver å velge komponenter via funksjonen ”enable mouse selection” på en trådmodell i AVAL, ville det vært en fordel

om dette fungerte mer som i AutoCAD. I AVAL kan man ikke velge interne komponenter, eller komponenter som ligger bak andre komponenter. Det er mulig å gjøre dette på andre måter, men det beste ville vært om det var mulig via ”enable mouse selection”.

- **Stort behov for data.** Dette er ikke en svakhet ved AVAL. Det er en fordel at AVAL kan ta hensyn til så mange effekter som mulig, og at brukeren har anledning til å spesifisere disse effektene som han selv vil. Men det betyr at man må belage seg på å bruke mye tid på å skaffe til veie gode data, blant annet om sårbarhet for alle komponenter som inngår i målmodellen.
- **Alle filer lagres som tekstfiler.** Dette er en stor fordel, og gjør det mulig for en bruker å manuelt gjøre endringer ved behov. Det er en veldig fleksibel måte å gjøre endringer på, men man må vite hva man holder på med. Programmet som leser tekstfilene kunne kanskje vært litt mer robust, og ikke gitt feilmelding dersom det kommer med et linjeskift for mye.

4 GENERERE VÅPENMODELLER

4.1 Våpeneditoren

Våpeneditoren i AVAL er oversiktlig og grei å bruke – dersom man setter seg inn i hva hver enkelt parameter betyr. Dette kommer ikke intuitivt frem i selve editoren, men er greit forklart i den skriftlige dokumentasjonen.

Et våpen i AVAL modelleres som en ”warhead carrier”, heretter kalt ”våpenet” i mangel av en bedre norsk oversettelse. Denne inneholder ett eller flere stridshoder (”warheads”) og ett sensorsystem (”fuze system”). Stridshodene kan inneholde en eller flere stridshodeeffekter (”warhead effects”), og sensorsystemet kan inneholde en eller flere sensorer/brannrør (”sensors”). Ordene i parentes er terminologien som benyttes i AVAL.

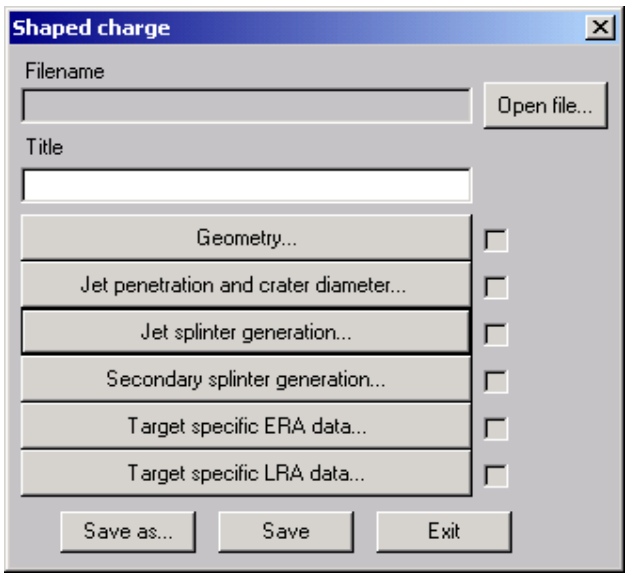
For oss har det vært mest aktuelt med tids- og anslagsbrannrør. Begge disse sensortypene er trivielle å generere. Plassering av sensorer i et sensorsystem har også gått veldig greit. Vi antar at laser- og magnetsensorer gjør situasjonen noe mer komplisert.

Når det gjelder våpen, er det generering av stridshodeeffekter som har vært mest omfattende for oss. Vi har også eksperimentert litt med rotering/forskyvning av en stridshodeeffekt i et stridshode, i motsetning til rotering/forskyvning av et stridshode i selve våpenet. Dette er beskrevet nærmere i kapittel 4.3.

For oss har det så langt vært tre stridshodeeffekter som har vært aktuelle å generere: Hulladninger, KE-prosjektiler og splintvåpen.

4.2 Hulladninger

Menyen for generering av hulladninger ser slik ut:



Figur 4.1 Meny for å generere hulladninger.

Den inneholder seks undermenyer som må fylles med data. Data for LRA (Local Reactive Armour) har vært av liten interesse for oss. Data for ERA (Explosive Reactive Armour) er absolutt interessant, men foreløpig har vi ikke laget verken mål eller våpen som håndterer ERA. Dette er blant de tingene som måtte kuttes ut på grunn av tids- og personellmangel i prosjekt 798. Vi kommer helt sikkert til å få bruk for dette senere. Noen refleksjoner rundt temaet har vi imidlertid gjort oss. Data for ERA og LRA er det eneste området der våpen og mål er avhengig av hverandre. Tilsynelatende kan man for et våpen kun oppgi effekten mot ERA til ett bestemt mål. Dette er lite fleksibelt. Det burde holde å låse effekten til en liste av ERA-moduler.

Geometrien beskriver kun hastigheten på jeten og hulladningens kaliber. ”Jet penetration and crater diameter” er menyen der man angir penetrasjonsdybde og kraterdiameter som funksjon av standoff. Dette er som regel mer eller mindre kjent for en bestemt hulladning. Man angir også standardavviket for penetrasjonsdybden (også dette som funksjon av standoff). Her ble det litt mer gjetting for vår del, men vi var ganske sikre på at standardavviket skulle være lite i forhold til penetrasjonen.

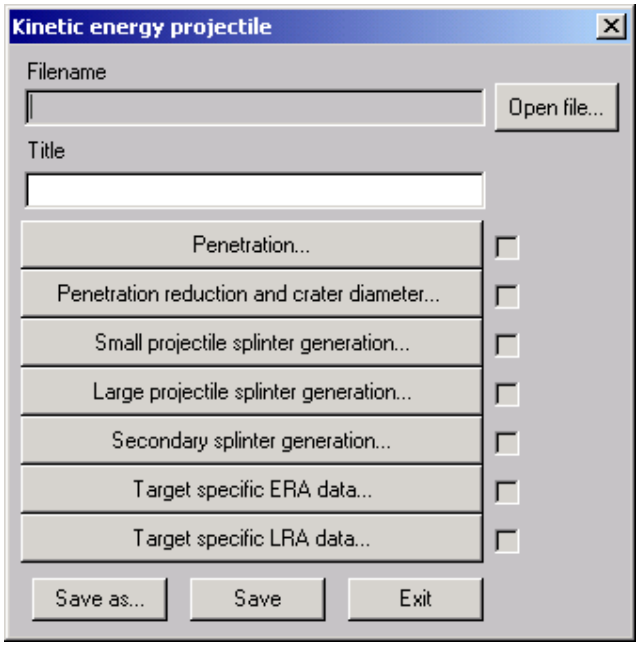
De to neste menyene omhandler såkalte ”Behind Armour Debris” – BAD. ”Jet splinter generation” beskriver primærsplintene som dannes etter perforering, og ”secondary splinter generation” beskriver sekundærsplintene. Dette beskrives i form av fordelingsfunksjoner for antall splinter, retningen splintene beveger seg i og penetrasjonsegenskapene til splintene. Disse dataene er av stor betydning, da det ofte er BAD som står for de fleste skadene i et kjøretøy – spesielt personellskader. Dette er dessverre et område der vi føler at vi ikke har veldig gode data. Noen av dataene kan anslås med tilstrekkelig nøyaktighet, men ikke alle.

Når det gjelder selve menyene for primær- og sekundærsplinter, så er de uklare på ett punkt, nemlig når det gjelder hvorvidt man ønsker å bruke en rektangulær fordelingsfunksjon eller ikke. Dersom man ønsker å bruke en annen fordelingsfunksjon, kommer det ikke klart frem av menyen hvorvidt denne nye fordelingsfunksjonen vil bli brukt på vinkelfordelingen, antallsfordelingen eller penetrasjonsevnefordelingen. I brukermanualen kommer det imidlertid frem at denne alternative fordelingsfunksjonen kun gjelder vinkelfordelingen. Man kan selvfølgelig se for seg muligheten for å spesifisere fordelingsfunksjonen for hver enkelt av de aktuelle egenskapene. Dette ville gjort programmet mer fleksibelt, men foreløpig har vi såpass lite datagrunnlag her at vi ikke ville vært i stand til å utnytte denne funksjonaliteten.

Når det gjelder selve ”shaped charge”-menyen (Figur 4.1), er det et problem at man ikke kan skru av ERA-dataene dersom man først har laget dem. Dette gjelder for så vidt også for KE-menyen, og også for LRA-data. Alle undermenyene har en liten boks til høyre for seg der det dukker opp en hake når man har fylt inn data. Det er imidlertid ikke mulig å fjerne denne haken igjen. For effekter som ikke er strengt nødvendige burde det være mulig å skru av og på disse effektene som man ønsker. For primær- og sekundærsplinter kan man skru av effekten inne i selve undermenyen. Det hadde imidlertid vært mer intuitivt også for disse effektene om man bare kunne fjerne avkryssingen til høyre for menyvalget, og dermed skru av effekten.

4.3 KE-prosjektiler

Menyen for å generere KE-penetratorer er ganske lik menyen for å generere hulladninger. Ett av unntakene er at for KE-penetratorer deles primærsplintene i to kategorier; store og små splinter. Dette kan være nødvendig fordi det i programmet ikke er noen kobling mellom antall, retning og penetrasjonsevne til splintene – alle tre parametrene trekkes uavhengig av hverandre. Når man deler splintene inn i to kategorier, kan man til en viss grad gjøre antall og retning avhengig av penetrasjonsevnen til splintene ved å sørge for at store splinter har større penetrasjonsevne enn små splinter, og så gjøre antallet og retningen til store og små splinter forskjellige.



Figur 4.2 Meny for generering av KE-prosjektiler.

I undermenyen *Penetration* spesifiserer man prosjektilets opprinnelig masse, samt dets penetrasjonsevne som funksjon av hastighet. I tillegg beskrives penetrasjonsreduksjon som funksjon av yaw og anslagsvinkel. I undermenyen *Penetration reduction and crater diameter* beskrives korreksjoner i pilens egenskaper ettersom den penetrerer et materiale. Splinter beskrives omtrent som for hulladninger.

Vi har så langt ikke fått lagt inn data for et ”reelt” KE-prosjektil, men det virker som om problemet også her er å skaffe data for dannelse av splinter etter perforering av panser (BAD). De andre dataene har vi bedre kontroll med.

Selv om vi ikke har lagt inn data for et ”reelt” KE-prosjektil, har vi eksperimentert litt med skråstilt pilammunisjon. Vi har tatt utgangspunkt i våpenet WA-100mmKE-0110 som følger med AVAL. Dette våpenet har vi så laget to nye varianter av: Én variant der effekten er skråstilt i stridshodet, og én variant der stridshodet er skråstilt i våpenet. Så har vi skutt med det opprinnelige våpenet og disse to variantene mot målet TA-Afv-0110, som følger med AVAL 4.2. Resultatene for de skråstilte våpnene ble svært like, og dårligere enn for det opprinnelige våpenet. Resultatene for de to våpnene ble ikke identiske, men forskjellen var så liten at den ikke var signifikant. At resultatene ikke ble helt identiske selv om vi brukte samme input-seed til randomgeneratoren for begge tilfellene, skyldes antakeligvis at selv om vi fikk den samme

rekken med tilfeldige tall, så er måten simuleringene håndteres på litt forskjellig. Dette betyr at tall nummer *n* i rekken ikke nødvendigvis har den samme funksjonen i de to tilfellene.

4.4 Nødvendige data

Som for generering av mål, er det også for generering av våpen tilgjengelighet på grunnlagsdata som er det største problemet. For mange våpen er de fleste parametrene heldigvis enkle å finne, men spesielt for generering av BAD; det vil si primær- og sekundærsplinter, har det ikke vært lett å skaffe gode data. Det er mulig at det finnes fysiske modeller som beskriver sammenhengen mellom våpendata og BAD, men vi har så langt ikke funnet noen slik enkel modell.
BAD er spesielt viktig for sårbarheten til personell, og da det i mange sammenhenger er nettopp dette som er interessant, blir det helt sentralt å skaffe gode data for BAD. Nok en gang kan ikke AVAL lastes for problemene med datainnsamling, det må sees på som en styrke at man kan variere disse parametrene som man ønsker

4.5 Splintvåpen

FFI-Prosjekt 837, *Airburst teknologiprogram*, ser på MP-ammunisjon for 30mm kanon, og skal utvikle en teknologidemonstrator for slik ammunisjon. Blant annet for å vurdere betydningen av usikkerheten til tidsbrannrøret, samt sammenligne effekten av forskjellige splintstørrelser, har AVAL vært et aktuelt verktøy. Ulike prefragmenterte varianter av denne granaten, der alle fragmenter har lik masse, er analysert for å bestemme hvilke fragmentstørrelser som gir størst virkning i målet.

Det er i hovedsak personell med ulik beskyttelsesgrad som er brukt som mål i dette prosjektet. Fordelen med å bruke fragmenter med lik masse i dette tilfellet er at sårbarhetsdataene blir riktigere når Sperazzas formel skal konverteres til ”AVAL-format” (se kapittel 6.1).

AVAL kan ta en fil generert av programmet *SplitX* og konvertere dette til en våpenfil for splintdata i AVAL. Det har imidlertid kommet en ny versjon av SplitX, og det ser ut til at formatet på filene blir endret i denne nye versjonen. En ny modell av AVAL må derfor ta hensyn til dette nye SplitX-formatet.

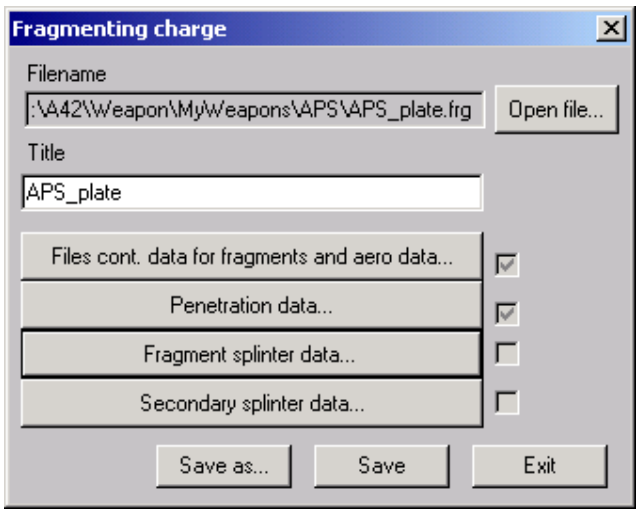
I prosjekt 798 har vi studert virkningen av aktive beskyttelsessystemer (APS). Man skiller mellom såkalte *hardkill-systemer* som prøver å fysisk ødelegge et innkommende missil, og *softkill-systemer* som prøver å narre sensorene til trusselen på et vis. I forbindelse med AVAL har det vært mest interessant å se på hardkill-systemer. Det finnes ikke mange ferdig utviklede slike systemer på markedet i dag. Det russiske *Arena*-systemet er det eneste hardkill-systemet som er deployert på stridskjøretøy i dag, men flere systemer er på tegnebrettet. Et APS består av en sensordel, en prosessor og en virkningsdel. Virkningsdelen i Arenasystemet er en boks som sender ut en skur av splinter. I AVAL har vi modellert en slik splintdannende plate som et splintvåpen.

Vi ønsker en plate som er kvadratisk og består av 14x14 splinter. Hver splint veier ca 1.5 gram og har en hastighet på omlag 2000 m/s. Vi kjenner posisjonen og retningen til hver splint, og ønsker å putte dette rett inn i våpenfila til AVAL. Ved å åpne en eksisterende fil og editere denne, virker det også som vi får til dette. Men vi kan ikke se at dette formatet er dokumentert

noe sted. I brukerveiledningen er det dokumentert hvordan formatet på forskjellige filer som genereres med SplitX eller FragM skal se ut. Det står også at filer basert på eksperimentelle data skal ha samme format som for FragM. Vi forstår det slik at alle de filformatene som er dokumentert i brukerveiledningen, er filer som AVAL kan lage sine egne filer fra. Vi savner en dokumentasjon av de filene som AVAL selv faktisk lager. For vårt tilfelle var det enkleste å generere denne fila direkte. Hvis vi har forstått det rett, er formatet på AVALs fil som følger:

Posisjonsvektor, retningsvektor, hastighet, masse, referanseareal

Totalt er det altså ni datapunkter pr splint, og vi har 14x14, altså nesten 200, splinter For oss var det enkleste å lage fila vår i Excel, for deretter å oversette denne fila til et tekstformat. Dette tekstformatet laget vi så en header til, og rettet de problemene som så ofte oppstår når man skal skifte mellom Excel- og tekstformat.

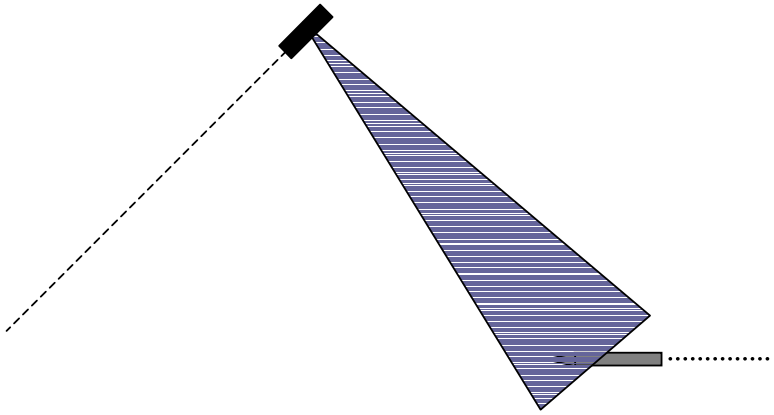


Figur 4.3 Meny for generering av fragmenterende våpenvirkning.

For aerofilen, som er en fil som beskriver hvordan splintene påvirkes når de farer gjennom lufta, brukte vi data fra en av de medfølgende filene. Sekundærsplinter har vi sett bort fra i denne sammenhengen. Når det gjelder penetrasjonsevne til splintene, hadde vi en god formening om denne. Så for dette formålet hadde vi ingen spesielle problemer med å skaffe de nødvendige data.

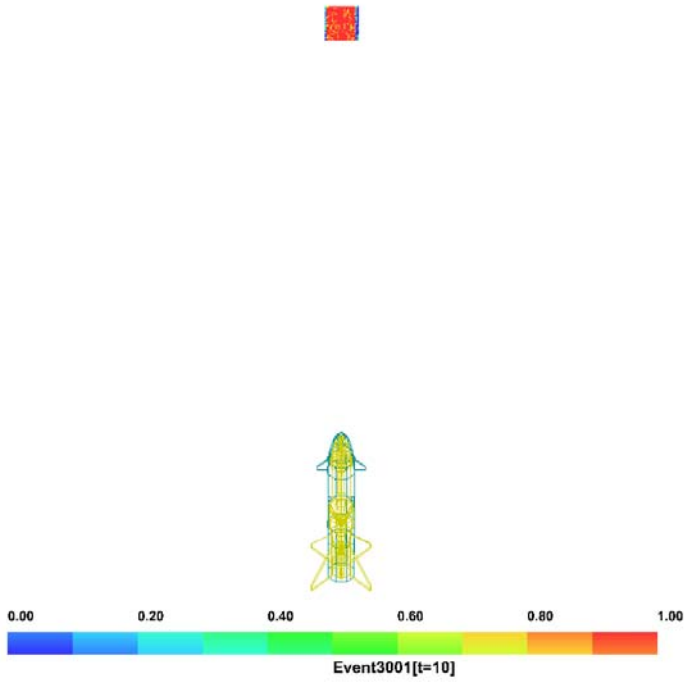
Som mål brukte vi antitankmissilet som fulgte med AVAL. Det står klart for oss at vi ville fått bedre resultater ved å lage vårt eget mål. Dette var også planen i utgangspunktet, men dette måtte utgå på grunn av personellmangel.

Vår splintvirkende kassett skyter ut splintene normalt på fartsretningen, se Figur 4.4



Figur 4.4 Kassetten skyter ut splinter normalt på fartsretningen

Denne situasjonen gir en sårbarhetsfigur som er litt spesiell. Hvis man ser i skyteretningen til kassetten, får man Figur 4.5. Det røde feltet angir her posisjonen der kassetten settes av. Det er ikke intuitivt enkelt å forstå denne figuren, men den er helt riktig, selv om men den ikke sier noen ting om sårbare områder på målet. Det den forteller, er hvor stridshodet må settes av for å få en bestemt effekt. At stridshodet settes av, betyr i denne sammenheng at det sendes ut en skur av splinter normalt på våpenets hastighet. For å oppnå en effekt må man derfor sette av stridshodet på en slik måte at *splintene* treffer målet, ikke selve våpenet.



Figur 4.5 Sårbarhetssimulering med en splintdannende kassett mot et missil

Noe som kunne vært interessant i denne sammenhengen, er hvor stor betydning det har at man bommer i tid, dvs at kassetten settes av for tidlig eller for sent. Det er ingen direkte måte å gjøre dette på, og dersom det var det, ville det gitt en merkelig, 3-dimensjonal ”sårbarhetsboks” som det ville vært svært vanskelig å tyde. Imidlertid kan man få til dette ved å skyte fra en annen vinkel, og endre retningen som splintene skytes ut fra kassetten med. Dette plottet viser

sårbarhet som funksjon av siktepunkt, og selv om siktepunkt egentlig er en 3-dimensjonal størrelse, er det bare hensiktsmessig å se på to dimensjoner samtidig.

4.6 Oppsummering

- **Data for ERA låses til ett bestemt mål.** Dette gjør bruken av våpenet lite fleksibelt. Det burde være mulig å låse effekten til bestemte ERA-moduler i stedet.
- **Man bør kunne "skru av" effekter i stridshodeeffektmenyen.** Det er ikke mulig å skru av effekten for ERA eller LRA hvis man først har klikket seg inn i disse menyene. Det burde være mulig å skru av en effekt ved å fjerne haken til venstre for den respektive effekten.
- **Formatet på splintvåpenfila som AVAL genererer er dårlig dokumentert.** Formatet på diverse filer som AVAL kan benytte for å lage denne filen, er dokumentert. Imidlertid finnes det tilfeller der man ønsker å lage denne filen direkte, og i slike tilfeller er det behov for at dette formatet er dokumentert.
- **Våpen som har en stridshodeeffekt som ikke går rett fremover, gir et lite intuitivt resultat i sårbarhetssimuleringer.** I et plott fra en sårbarhetssimulering er effekten av et skudd plottet i det punktet man sikter på. For våpen der stridshodeeffekten ikke går rett fremover, sikter man ikke rett på målet, men over eller ved siden av våpenet. Ønsker man å finne sårbare områder på målet, bør man lage seg et nytt våpen der stridshodeeffekten og våpenets bevegelsesretning er parallelle.
- **BAD-data er vanskelig å oppdrive.** Generelt er det enklere å skaffe data for våpen enn for mål. Problemet med våpen er data for BAD. Dette er selvsagt ikke et problem med AVAL, men et generelt problem ved beregning av slike effekter.
- **Oversiktlig våpeneditor.** Menyen for å bygge opp et våpen er enkel å bruke og manøvrere i. Vi har primært benyttet hulladninger og KE-prosjektiler, og menyene for disse stridshodeeffektene er gode og oversiktlige.

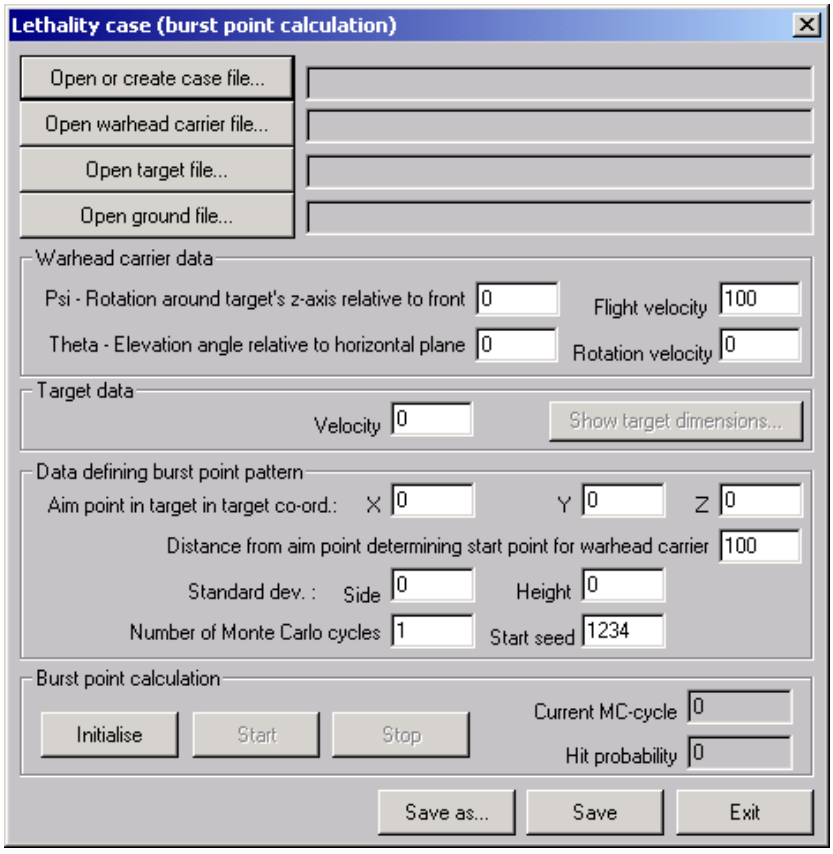
5 GJENNOMFØRING AV SIMULERINGER

5.1 Enkeltskudd

Simulering av enkeltskudd er velegnet for å teste et våpen eller mål som er under utvikling, eller for å studere enkelteffekter. Det er ikke egnet for å skaffe reelle resultater. En simulering av enkeltskudd er som regel rask å gjennomføre, og resultatet kommer umiddelbart. Resultatet kan enkelt visualiseres i AVAL-vinduet ved at komponenter som er utslått blir markert med rødt. Det kan riktignok være litt vanskelig å se hvilke komponenter som er slått ut hvis målet er komplisert og består av mange deler, men stort sett fungerer dette veldig greit. Uansett lagres resultatene i en tekstfil, så dersom det ikke er lett å se hva som har skjedd grafisk, kan man benytte denne. Man kan også se hvilken effekt som slo ut en komponent (penetrasjon, varme, trykk etc). Dette er nyttig når man holder på å lage utvikle en målmodell og skal undersøke om effekter som brann og detonasjon virker fornuftig i modellen. Funksjonen ”Time Control” kan benyttes for å se utvikling av røyk, trykk, varme og skade som funksjon av tid.

Generelt fungerer dette meget bra. Det går raskt å sette opp en simulering av enkeltskudd, og det er lett å se hva som har skjedd i ettertid.

5.2 Effektsimulering

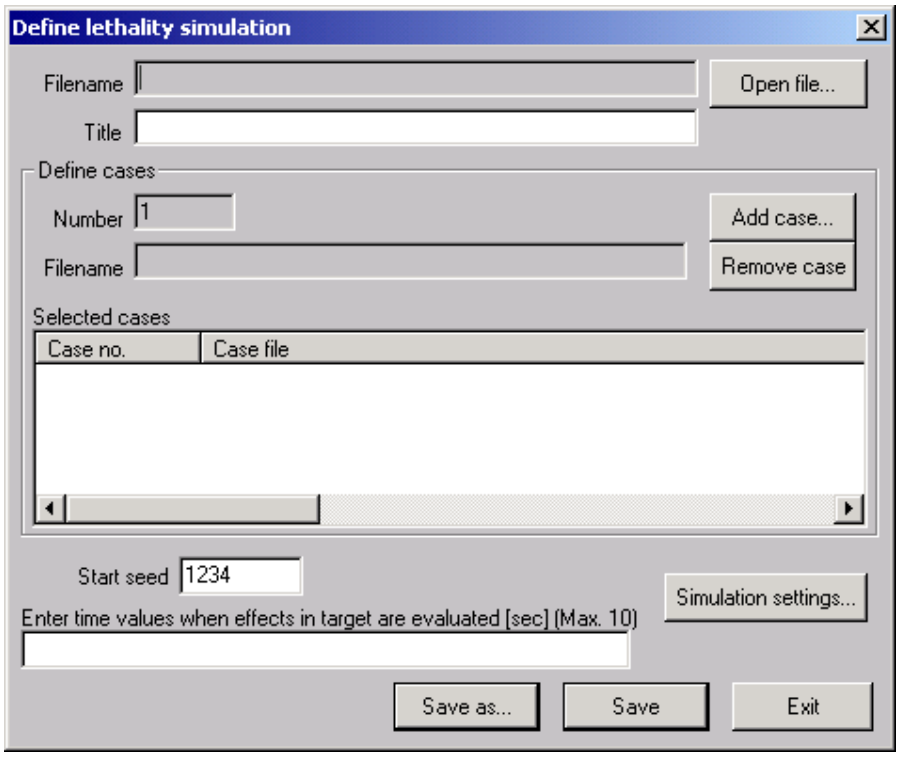


Figur 5.1 Vinduet for generering av casefil for effektsimuleringer.

Hvis man er ute etter å anslå med hvilken sannsynlighet et gitt våpen har for å slå ut et bestemt

mål, er det effektsimuleringer som er tingen. I AVAL starter man med å definere en ”case”. Vinduet for generering av casefil er vist i Figur 5.1. En casefil inneholder informasjon om hvilket våpen og hvilket mål som inngår i simuleringen. Anslagsvinkel, hastighet til våpen og mål, samt usikkerhet i treffpunktet angis også her.

Når man trykker på ”start”, genereres et sett med treffpunkter. En annen ting som spesifiseres i casefila er et ”input seed”. Dette er et tall som randomgeneratoren i AVAL bruker for å generere tilfeldige tall (tilfeldige tall generert av en computer kalles egentlig pseudotilfeldige tall, men vi går ikke nærmere inn på dette her). Et bestemt input seed gir en bestemt rekke med tilfeldige tall. Dette betyr at dersom man gjentar simuleringen med samme input seed, får man de samme treffpunktene. Dette kan være nyttig i tilfeller der man er interessert i å reprodusere et resultat.



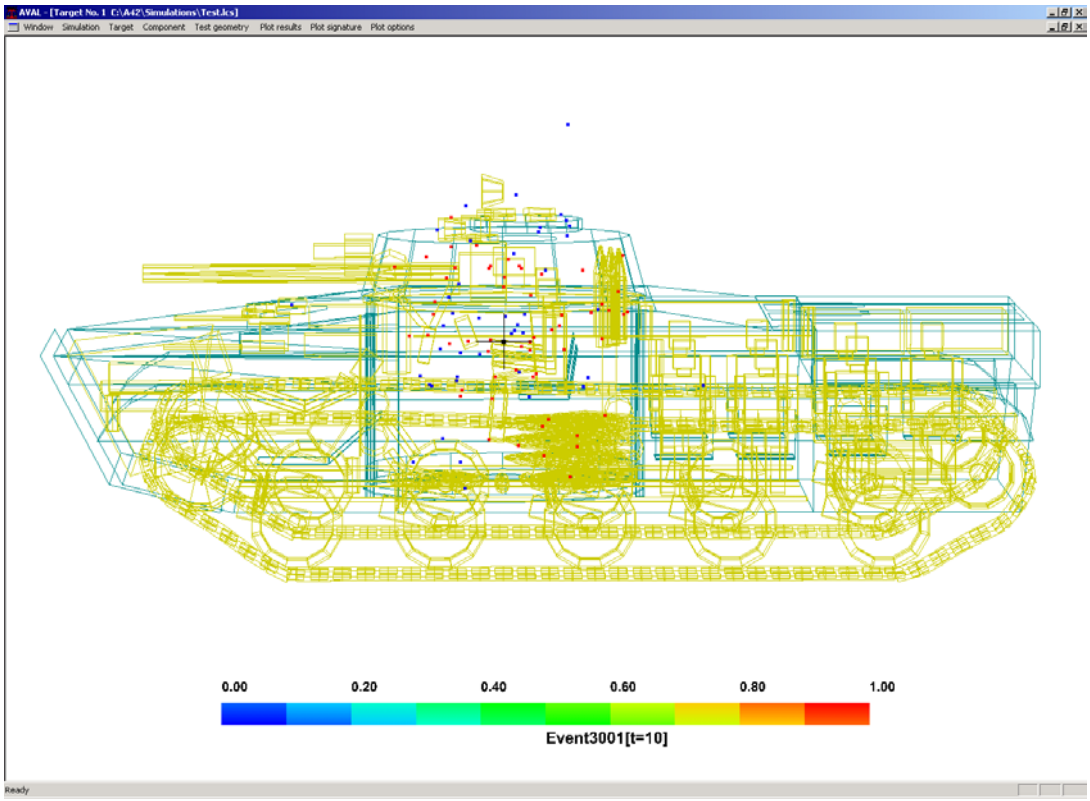
Figur 5.2 Vinduet for å definere effektsimuleringer.

Når man er ferdig med å spesifisere en case, går man videre med å definere en sårbarhetssimulering (se Figur 5.2). Her spesifiseres én eller flere casefiler, tiden man skal simulere til og spesielle settinger for simuleringen, ”simulation settings”. Alle casefilene må være mot samme mål og bestå av like mange simuleringer, men ellers kan de være forskjellige. Vi har ikke sett nytten av å kunne spesifisere flere casefiler i én simulering. Vi forstår heller ikke fullt ut hvordan dette fungerer, og det er ikke tilstrekkelig godt beskrevet i brukermanualen. De gangene vi har forsøkt dette, får vi bare plottet resultatet fra én av casene etterpå. Ved simulering av en enkelt case har vi ikke hatt problemer.

En liten ulempe med dette vinduet (Figur 5.2) gjelder når man skal fjerne en case. For å gjøre dette, må man først merke den. Dette er kun mulig ved å klikke på nummeret til casen, under ”case no.”. Klikker man på navnet, under ”case file”, blir ikke casen merket. Dette er en detalj, men bør like fullt endres.

Etter å ha definert simuleringen, kan man kjøre den. Det er mulig å kjøre flere simuleringer i serie, under ”simulation – consecutive”. Dette er lurt, da man gjerne ønsker å bruke tiden til å sette opp og definere scenarier, og kan kjøre simuleringene samlet over natten eller i helgene.

Etter å ha kjørt simuleringene, kan man hente ut resultatene enten grafisk eller direkte fra en tekstfil. Tekstfila inneholder det man trenger av informasjon, mens den grafiske fremvisningen gjør det mulig å se hvor man har truffet i de forskjellige simuleringene, samt hvilke av disse treffene som har gitt god uttelling. For effektsimuleringer er det tekstfila som gir den viktigste informasjonen. Et eksempel på grafisk fremstilling er vist i Figur 5.3.



Figur 5.3 Grafisk fremstilling av resultatene fra en effektsimulering.

En kuriositet her er skalaen under bildet. Fargene angir sannsynligheten for at en bestemt hendelse har inntruffet, gitt et treff i det aktuelle punktet. Men i en effektsimulering simuleres effekten i hvert treffpunkt kun én gang, slik at for hvert punkt har nødvendigvis en hendelse enten inntruffet eller ikke inntruffet. Således angir en blå prikk at hendelsen ikke har inntruffet, mens en rød prikk angir at hendelsen **har** inntruffet. Skalaen under bildet gir altså ingen mening, da verdiene null og én er de eneste mulige verdier.

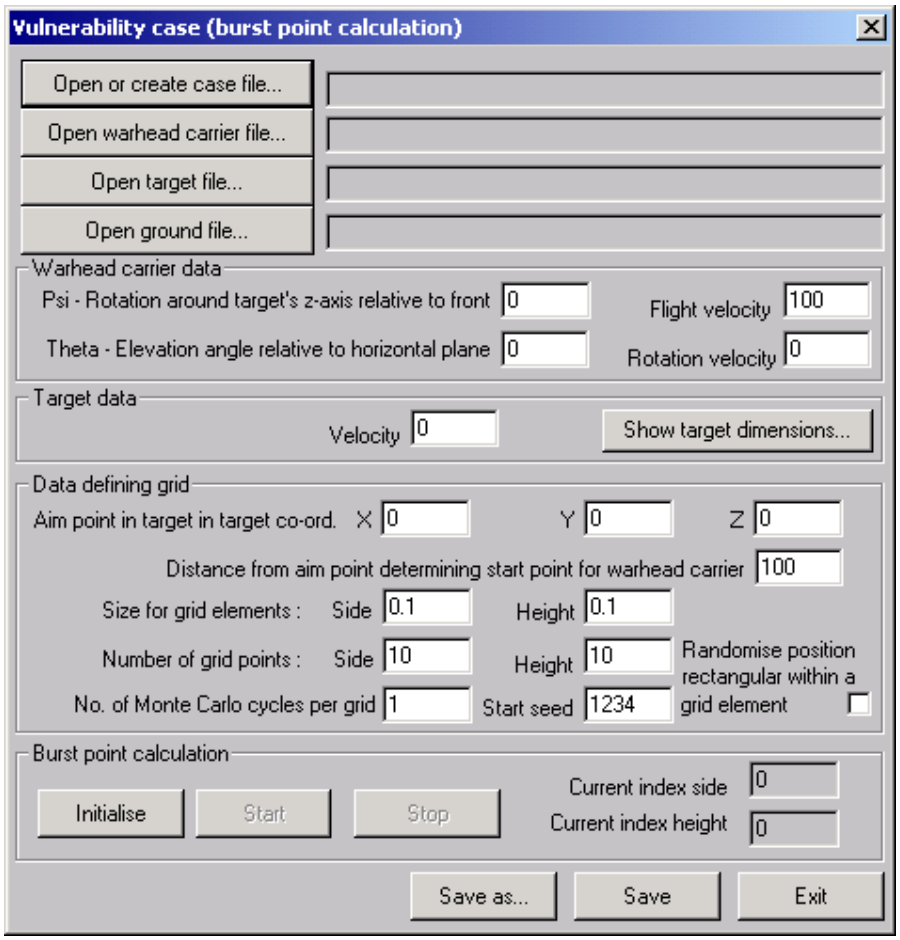
Når man leser en casefil i det grafiske vinduet, kommer målet fra denne casefilen automatisk opp på skjermen. Dette er bra. Men det kommer alltid opp slik at man ser målet i y-retningen. Dette er ikke så bra. Når man leser en casefil, bør målet automatisk komme opp slik at man ser det fra den retningen man skyter. Punktene i Figur 5.3 gis nemlig posisjonen der våpenet treffer målet, eventuelt i et bestemt plan dersom våpenet bommer på målet. Dersom man ser målet fra gal vinkel, blir det grafiske resultatet svært misvisende. Slik det er i dag, må man selv stille inn ”kameraet” slik at man ser målet fra riktig vinkel. Vi ser ingen grunn til at man skal kunne rotere målet i det hele tatt når man holder på med grafisk fremvisning av simuleringsresultater, så det

beste ville kanskje vært at målet var låst i en vinkel slik at man ser det fra den vinkelen det blir skutt. Muligheten til å forstørre/forminske målet, samt flytte målet rundt på skjermen, bør fortsatt være til stede.

5.3 Sårbarhetssimulering

Det kan virke litt rart å skille mellom effektsimuleringer og sårbarhetssimuleringer, ettersom begge deler jo simulerer effekten av et bestemt våpen mot et bestemt mål. Det er imidlertid gode grunner til å ha et slikt skille.

Som for effektsimuleringer er første skritt å definere en case. Vinduet for definering av casefil ser litt annerledes ut for sårbarhetssimuleringer enn for effektsimuleringer, se Figur 5.4. Data for våpen og mål angis som for effektsimuleringer, men i stedet for siktepunkt og spredning angir man et grid som det skal skytes i. I tillegg angir man hvor mange skudd som skal skytes i hvert gridrektangel. Om man haker av boksen for ”randomise position rectangular within a grid element” skytes det ikke i senteret av gridrektangelet, men det trekkes tilfeldig hvor i dette rektangelet det skytes. Ved flere skudd mot hvert gridrektangel er det en fordel å benytte denne opsjonen.



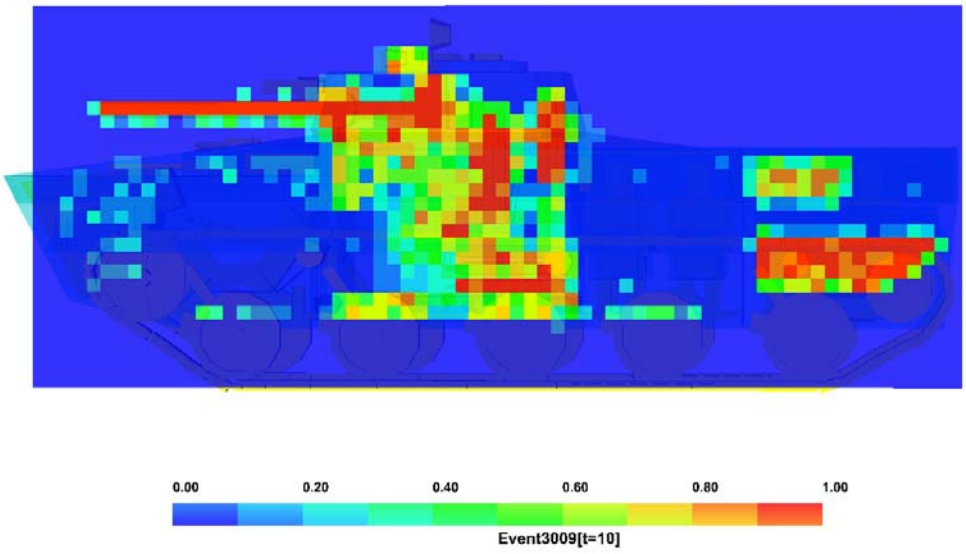
Figur 5.4 Vindu for generering av casefil til sårbarhetssimuleringer.

Måten selve gridet genereres på er litt ugunstig. Man spesifiserer senteret i gridet, antall gridrektangler og størrelsen på gridrektangelet. Det ideelle ville vært, etter at man hadde spesifisert skyteretningen, om man kunne få opp et bilde av målet sett fra denne vinkelen, og så

kunne man spesifisere nedre venstre og øvre høyre hjørne av gridet, samt antall gridrektangler i x- og y-retning. Det går for så vidt greit slik det gjøres i dag, men det tar litt tid å plassere gridet, og ofte blir det ikke helt slik man hadde forestilt seg. Siden en sårbarhetssimulering kan ta relativt lang tid, ønsker man minst mulig grad av prøving og feiling. Valget "show target dimensions" er til en viss hjelp, men ikke så mye hvis man skyter fra en ukurant vinkel.

Etter at man har laget casefilen, er neste skritt å definere en simulering, som for effektsimuleringer. Vinduet der man gjør dette er lik vinduet for definering av effektsimuleringer, med det unntaket at det kun er mulig å velge én casefil. Man definerer tider det skal simuleres til og spesielle settinger, akkurat som for effektsimuleringer. Etter å ha definert simuleringen, er det bare å gå i gang med selve simuleringen.

Resultatene fra sårbarhetssimuleringer kan også hentes frem grafisk eller i tekstfil. I motsetning til for effektsimuleringer, er det for sårbarhetssimuleringer svært verdifullt med den grafiske fremvisningen. Her kan man se hvilke områder på målet som er mest sårbare. Et eksempel på en grafisk fremstilling av en sårbarhetssimulering er vist i Figur 5.5. Her er gridrektanglene ti cm i x- og ti cm i z-retning (y-retningen er innover i arket/skjermen).



Figur 5.5 Grafisk fremstilling av resultatene fra en sårbarhetssimulering.

Grafisk fremvisning av resultatene fra sårbarhetssimuleringer har det samme problemet som for effektsimuleringer, nemlig at målet vises sett fra y-retningen når man leser casefilen. Resultatene tegnes i et plan bestemt av siktepunktet og skyteretningen, og dersom man ser målet fra gal vinkel, får man et feilaktig inntrykk av sårbarheten til målet. Ettersom den grafiske fremstillinger en mye viktigere når det gjelder sårbarhetssimuleringer enn når det gjelder effektsimuleringer, er dette et enda større problem for sårbarhetssimuleringer enn for effektsimuleringer.

5.4 Logging av resultater

Når man har definert en simulering og skal sette den i gang, kan man definere hvilke resultater som skal logges. For oss virker det som om man her har alle mulighetene man trenger. Vi skal ikke si så mye om dette, bortsett fra at vi setter pris på at man ikke trenger å logge all informasjon hvis man ikke ønsker det, da det tar lengre tid å foreta en simulering dersom man ønsker å logge all informasjonen.

5.5 Oppsummering

- **Funksjonaliteten med flere casefiler i samme simulering virker unødvendig.** Vi ser ikke hvilken fordel man har av denne funksjonaliteten. For øvrig er måten man fjerner casefiler fra simuleringen på litt klønete. Man må klikke på nummeret foran fila; det burde også gå an å klikke på selve filnavnet.
- **Skalaen under figuren for effektsimuleringer har ingen mening.** I effektsimuleringer vil, for hvert enkelt treffpunkt, en hendelse enten ha inntruffet eller ikke inntruffet. Sannsynligheten for en hendelse vil med andre ord enten være null eller én, aldri en mellomting.
- **For plotting av resultater bør kameravinkelen være lik skyteretningen.** Dette kan man selvsagt stille inn selv etter at man har åpnet figuren, men det ville vært en fordel om dette var default kameravinkel. Dette gjelder både effektsimuleringer og sårbarhetssimuleringer, men ikke simulering av enkeltskudd.
- **Spesifisering av grid for sårbarhetssimuleringer kan forbedres.** Vårt forslag er at etter at man har spesifisert skyteretning, kan man få opp et bilde av målet sett i skyteretningen. Så kan man plassere et rektangel i dette bildet, og angi antall gridrektangler i side og høyde.
- **Grafisk fremstilling av resultatene fra sårbarhetssimuleringer er god,** i alle fall dersom våpeneffekten går i samme retning som våpenet. For andre tilfeller må man holde tunga rett i munnen og vite hva man driver med.
- **Simulering av enkeltskudd er enkelt og effektivt.** Det går raskt både å sette opp og kjøre slike simuleringer. Det er lett å se i etterkant hva som har skjedd.
- **”Consecutive simulations” er en god funksjonalitet.** Det er veldig greit å kunne definere et sett med scenarier, og så starte alle samtidig.
- **Brukerbestemt inputseed til randomgeneratoren kan være en fordel.** Det er riktignok ikke spesielt ofte man har behov for det, men enkelte ganger kan det være praktisk å kunne kjøre en simulering en gang til med den samme rekka av ”tilfeldige” tall.
- **Det er bra at man kan velge hvilke resultater som skal logges.** Det sparer tid å ikke måtte logge unødvendige detaljer i enkle simuleringer, samtidig som man har muligheten til å studere alle parametre i de tilfellene der er nødvendig.

6 VIRKNINGSMODELLER I AVAL

6.1 Modell for sårbarhet av komponenter

AVAL antar at sårbarheten til en komponent kun er avhengig av dybden på penetrasjonskanalen gjennom den. Dette betyr at det ikke tas hensyn til diameteren til krateret, og at en hulladning, pil, stor og liten splint som penetrerer samme dybde i en komponent, har samme sannsynlighet for å ødelegge komponenten. For hulladningen/pilen/splinten man har tatt utgangspunkt i ved beregning av sårbarhet, får man selvsagt riktig resultat likevel. Det er når man ønsker å benytte et annet våpen mot dette målet, for eksempel en splint med større masse, at problemet oppstår.

Sperazzas formel (1) sier at sannsynligheten for skade ved treff av en splint med masse m og hastighet v er gitt ved

$$P = 1 - \exp\left[-a \cdot \left(m \cdot v^{3/2} - b \right)^{n} \right]$$

der parametrene a, b, og n er avhengige av den taktiske situasjonen og hvor lang tid det skal gå før skaden gjør seg gjeldende. Denne formelen gir ikke samme sannsynlighet for skade for splinter med forskjellig masse, selv om de penetrerer like dypt i soldaten.

Vi har forstått det slik at i fremtidige versjoner av AVAL vil sårbarheten til en komponent være avhengig av *volumet* som eroderes fra en komponent – ikke lenger *dybden* av penetrasjons-kanalen. Dette vil muligens gi større overensstemmelse med Sperazzas formel. Imidlertid kan vi ikke forstå annet enn at en slik endring må medføre at penetrasjonssårbarhetsfilen til alle eksisterende mål må oppgraderes dersom disse målene skal kunne benyttes i fremtidige AVAL-versjoner. Det dreier seg bare om én fil, men det er mange data som må skaffes. Det er selvsagt likevel en fordel at modellen endres for å gi mer realistiske resultater.

6.2 Miner

Tradisjonelle miner, altså miner som ligger på eller under bakken og eksploderer, er ikke egnet for simulering i AVAL. Riktignok kan hulladnings- eller splintdelen av kjøretøyminer simuleres, men effekten av trykk kommer ikke med. Eksplosjonen av en mine vil føre til at et kjøretøy kastes oppover. Spesielt vil komponenter som er plassert på kjøretøyets gulv kunne ta skade av dette. Soldater som har føttene på gulvet i det en mine går av, vil få ødelagt bena, og muligens bli drept. Selv om AVAL har metoder å beregne trykkutvikling og skader som funksjon av trykk på, beregnes ikke forplantningen av trykk gjennom kjøretøyet.

For FFI er dette en viktig faktor å ta hensyn til, da miner ofte kan være hovedtrusselen for kjøretøy i internasjonale operasjoner. Vi har derfor anskaffet en egen modell, *MADYMO*, for å studere slike effekter. Det ville selvsagt vært ønskelig med én felles sårbarhetsmodell for alle effekter, inkludert miner, men vi antar det ville bli svært omfattende å utvide AVAL til også å ta

hensyn til slike effekter.

6.3 Trykk

AVAL håndterer trykkvirkning fra ladninger som detoneres. Vi har ikke sett på hvordan denne modellen fungerer, men antar at det håndteres på en tilfredsstillende måte. Det vi imidlertid har lagt merke til, er at trykk ikke får ladninger til å detonere. Dette er i tråd med hva vi mener er realistisk. Imidlertid kan all ammunisjon settes av ved treff i en enkelt granat. Modellen som håndterer mulig massedetonasjon når en ladning i en ammunisjonsgruppe detonerer, skal håndtere dette. Det mest "korrekte" ville muligens vært å modellere ammunisjon som et eget våpen som kan settes av, og at splinter fra dette våpenet igjen avsatte annen ammunisjon, men det er antakeligvis unødvendig tungvint. Når ammunisjonen i et kjøretøy detonerer, får man som oftest en "catastrophic kill" uansett. Det er neppe nødvendig å gjøre det mer komplisert enn som så. For fartøy og andre måltyper kan selvsagt situasjonen være en annen.

En trykkeffekt som tilsynelatende ikke er modellert, er trykket som oppstår når en KE-penetrator eller en hulladning penetrerer panseret rundt et volum. I virkeligheten oppstår det i slike situasjoner et trykk som kan være skadelig for personell. Nå er det riktignok BAD som antakeligvis står for de mest omfattende personellskadene, men trykkvirkningen kan også være av betydning, spesielt for kjøretøy med spall-liner, der effekten av splinter reduseres.

6.4 Splintdannelse

Når den ytterste veggen i et mål perforeres, dannes det BAD bak denne dersom flaten er satt til å være splintdannende. Dette er viktig. Men dersom virkningen perforerer en vegg nummer to, dannes det ikke splinter bak denne, selv om den er satt til å være splintdannende. Dette kan i noen tilfeller være av stor betydning, og må tas hensyn til når man genererer mål.

Vi forstår det slik at det i neste versjon av AVAL vil være slik at alle flater kan være splintdannende, og dette problemet blir da løst.

6.5 Brann/røyk

Både brann, brannslukking og røykutvikling tas hensyn til i AVAL. Dette er effekter vi dessverre ikke har fått tid til å se så nøye på som vi hadde håpet. En styrke ved AVAL er imidlertid at man ikke trenger å bruke disse effektene. Man kan bygge mål og kjøre simuleringer der kun penetrasjon og perforasjon tas hensyn til. Så kan man senere legge til effekter fra brann, røyk, trykk og andre ting.

6.6 Tekstfiler

De fleste filene som AVAL bruker, både for mål, våpen og i forbindelse med simuleringer, lagres i tekstformat. Dette gir brukeren muligheten til å gå inn og gjøre endringer direkte i fila. Slike endringer krever selvsagt at brukeren har god kjennskap til programmet, og at han kjenner formatet på de forskjellige filene. Selv minimale avvik fra dette formatet vil føre til

feilmeldinger.

Rutinene for innlesing av tekstfiler burde vært laget mer robust. Blant annet er det slik at et ekstra linjeskift på slutten av fila er nok til at programmet ikke kjører. Direkte editering i tekstfiler er svært verdifullt, og for enkelte filer også helt nødvendig. Imidlertid kan det av og til snike seg inn et linjeskift eller et mellomrom i filene, og dette bør programmet kunne håndtere. Til en viss grad kan man legge inn kommentarer i filene, men rutinene for innlesing av filer kan med fordel forbedres.

6.7 Egne modeller

AVAL har, i likhet med de fleste kommersielle programmer, en lukket kildekode. Det innebærer at det ikke er mulig for en bruker å gjøre endringer eller lage tillegg til programmet. Dersom man har utviklet egne modeller for bestemte fenomener, er det altså ikke mulig å putte disse modellene inn i AVAL selv.

De fleste fenomener er beskrevet ved hjelp av tabeller og parametre som brukeren kan spesifisere. Dette hjelper en god del, men i enkelte tilfeller kan brukeren ønske å legge inn en helt egen beskrivelse av et bestemt fenomen.

6.8 Oppsummering

Her kommer de viktigste punktene fra kapittel 6.

- **Sårbarheten til personell er tvilsomt modellert.** Sårbarheten til komponentene er kun avhengig av hvor lang penetrasjonskanalen gjennom den er. Hvis tabellen er kalibrert for å være riktig for en splint på ett gram, blir den gal for en splint på ti gram.
- **Tradisjonelle miner kan ikke modelleres i AVAL.** Det finnes andre programmer for slike oppgaver. Ved FFI har vi valgt modellen *MADYMO*.
- **Trykk som oppstår når veggen i et volum perforeres av en pil/hulladning, modelleres ikke.** Det er ikke sikkert at denne effekten er spesielt stor, men det hadde vært en fordel om den ble tatt hensyn til.
- **Egne modeller kan ikke legges inn.** Slik vil det være i de fleste tilfeller for programmer med lukket kildekode.

7 ANDRE OBSERVASJONER

7.1 Grafikk i AVAL

Når vi på vår maskin [Compaq Armada med PIII 1GHz prosessor og 256 MB RAM] bruker full grafikkakselerator, fungerer ikke grafikken i AVAL. Dette er for så vidt ikke noe stort problem, da det fungerer når vi skrur ned ytelsen til grafikkortet. Imidlertid er det et lite irritasjonsmoment, da man gjerne ønsker full ytelse på grafikkortet i andre sammenhenger, og derfor må endre innstillinger avhengig av om man skal bruke AVAL eller andre programmer. Det er også viktig å opplyse om dette for nye brukere, da det slett ikke er trivielt å forstå at man må endre innstillingene på grafikkortet for at ting skal fungere normalt.

7.2 Beregningstider

Tiden det tar å simulere et enkeltskudd er som oftest under ett sekund, altså nærmest helt neglisjerbar. En effektsimulering innebærer at man først genererer et antall punkter basert på oppgitt spredning, og deretter simulerer effekten ved treff i disse punktene. Det er sjelden behov for mer enn tusen simuleringer. Genereringen av tusen punkter tar i størrelsesorden 5 sekunder. Å gjennomføre selve simuleringene tar noe lengre tid. Når vi skyter med våpenet WA-10mmSC-0110 mot målet TA-Afv-0110, som begge fulgte med AVAL 4.2, tar det omlag ett og et halvt minutt å foreta tusen simuleringer.

Det store tidsforbruket kommer når man skal gjennomføre sårbarhetssimuleringer. En stridsvogn er som oftest over fem meter lang og over to meter høy. Hvis man skyter på stridsvognen fra siden og ønsker en oppløsning på én cm i begge retninger, blir dette over hundre tusen gridrektangler. I tillegg ønsker man å skyte flere skudd i hvert gridrektangel for å få sårbarheter mellom null og én – hundre skudd i hvert gridrektangel gir et godt resultat. Hvis man ønsker tilfeldige treff innenfor hvert gridrektangel, betyr dette at det skal genereres over én million treffpunkter og utføres like mange simuleringer. Slike simuleringer kan ta flere dager. Heldigvis kan man la simuleringene kjøre over natten eller over en helg. Det er nesten verre at det tar så lang tid å generere treffpunkter (”burst point calculation”). Det burde etter vår oppfatning ikke være nødvendig at dette tar så lang tid som det gjør. Man må vente til denne er ferdig før man kan starte selve simuleringen. Vi kjørte et testscenario med 68x28 gridrektangler, og 100 simuleringer i hvert rektangel. For denne simuleringen tok det omtrent elleve minutter å generere treffpunkter. Her var hver gridrektangel 10 cm x 10 cm. Med en oppløsning på 1 cm x 1 cm vil det ta 100 ganger så lang tid. Det er uvisst hvorfor dette tar så lang tid, men når det nå en gang er slik, burde man ha mulighet til å gjøre dette som en integrert del av simuleringen, slik at det ikke blir behov for input fra brukeren mellom genereringen av treffpunkter og selve simuleringen. Dette er imidlertid ingen stor sak.

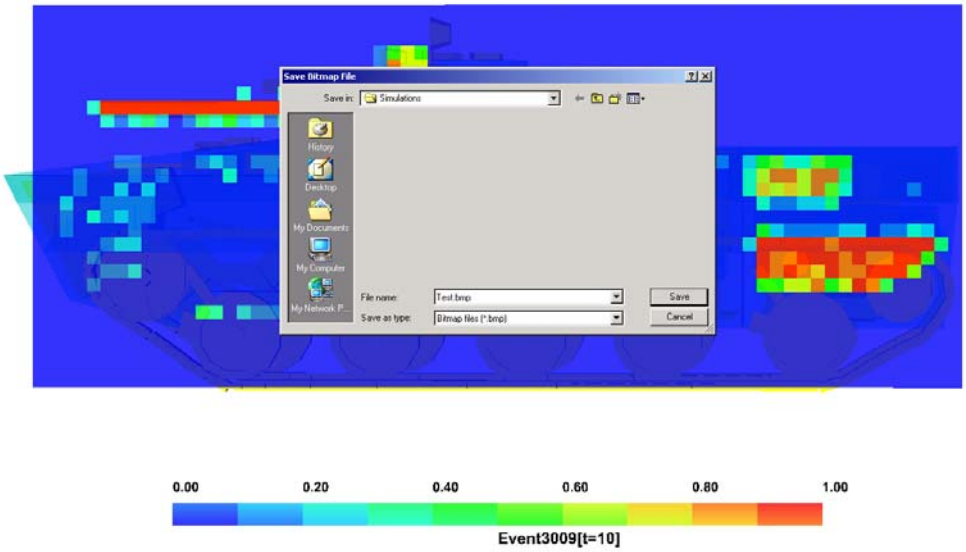
7.3 Store simuleringer

Som nevnt i kapittel 7.2 ønsker man av og til å kjøre omfattende sårbarhetssimuleringer, gjerne med en oppløsning på én cm. Det virker ikke som AVAL håndterer slike simuleringer godt. Vi får alltid generert treffpunkter, selv om det tar svært lang tid (i ett tilfelle tolv timer), men vi får

ikke kjørt simuleringen. Når vi skal definere simuleringen, og prøver å spesifisere hvilken casefil som skal brukes, henger maskinen seg. Etter en tid får vi en ”out of memory”-feilmelding. Det er kanskje forståelig, selv om det synes merkelig at maskinen klarer å generere casefilen, men ikke lese den. Vi forventer ikke at AVAL skal kunne lese filer av enhver størrelse inn i minnet, men det kan finnes andre løsninger. For eksempel kan AVAL gjøre et overslag over størrelsen på casefila før den genererer denne, og dele fila opp i flere mindre filer. AVAL bør i alle fall kunne håndtere dette på en eller annen måte, eller i det minste gi en advarsel når man prøver å starte en enorm simulering.

7.4 Lagre bilder

AVAL har funksjonalitet for å eksportere bilder til en bitmap-fil. Dette er gunstig med tanke på presentasjon i for eksempel Powerpoint eller Word. Et lite problem med denne funksjonaliteten er vist i Figur 7.1. Her kan man se at dialogboksen for eksportering av figur har kommet med på figuren. Det burde den helt klart ikke gjort.



Figur 7.1 Et bitmap-bilde eksportert fra AVAL.

AVAL har også en annen måte å gjøre dette på, nemlig ”Copy to clipboard”. Innholdet i vinduet lagres da som en figur i utklippstavlen, og kan limes inn i for eksempel Word eller Powerpoint. Dette går raskere, og man slipper problemet med at dialogboksen følger med.

7.5 Dokumentasjon og kurs

Dokumentasjonen til AVAL er delt i tre: En brukerveiledning (”User Manual”), en målbeskrivelsesmanual (”Target Description Manual”) og en teknisk manual (”Reference Manual”). Brukerveiledningen beskriver hvordan man installerer programmet og hvordan man utfører simuleringer. Målbeskrivelsesmanualen beskriver hvordan man bygger opp målmodeller og gir dem de nødvendige egenskaper. Her kommer også en beskrivelse av AVAL-CAD inn.

Den tekniske manualen beskriver det man trenger for å lage våpenmodeller, og inneholder også informasjon om hvordan AVAL fungerer, altså hvordan forskjellige effekter tas hensyn til.

Brukerne av AVAL ved FFI har deltatt på kurs i bruken av AVAL. Kursene var delt i fire deler. Det første kurset omhandlet bruken av AVAL, der man lærte å utføre simuleringer. Ett kurs handlet om generering av våpenmodeller, og to kurs handlet om generering av målmodeller (ett tok for seg geometrien, mens det andre tok for seg øvrige parametre ved målet).

Det er ganske enkelt å utføre simuleringer i AVAL, så ettersom vi har vært på kurs har vi hatt liten bruk for brukerveiledningen. Vi klarte imidlertid ikke å finne ut hva det innebærer å sette opp flere casefiler i én enkelt effektsimulering. Brukerveiledningen beskrev hvordan dette kan gjøres, men ikke hva det innebærer.

Målbeskrivelsesmanualen har heller ikke vært flittig brukt. Vi har tatt et eget kurs i AutoCAD for lettere å kunne generere målgeometrier. AVALCAD er stort sett grei å bruke uten at man stadig behøver å titte i manualen. Når det gjelder øvrige parametre, har manualen gitt svar på de spørsmål vi har forventet at den skulle kunne svare på.

Den delen som helt klart har vært mest brukt, er den tekniske manualen (”Reference Manual”). Vi har opplevd denne som god, og dekkende for våre formål. Den beskriver hvordan de fysiske modellene i AVAL fungerer, og hva de forskjellige parametrene betyr. Etter hvert som vi kommer til å bruke AVAL på mer avanserte måter, forventer vi å få en bedre formening om hvor god dokumentasjonen faktisk er. Det eneste vi savner så langt, er en dokumentasjon av filen som AVAL genererer i forbindelse med splintvåpen, se kapittel 4.5.

7.6 Oppsummering

Her kommer de viktigste punktene fra kapittel 7.

- **Skjerminnstillinger må endres for at grafikken skal fungere godt.** Dette bør det være mulig å endre.
- **Feil i eksportering av bitmapfil.** Her kommer menyvinduet med på bildet.
- **Korte beregningstider.** AVAL må sies å være temmelig rask. Det eneste vi har noe å utsette på her, er at det tar litt for lang tid å generere treffpunkter i sårbarhetssimuleringer. Vi tror det er mulig å gjøre dette raskere. Er det ikke det, ville det vært en fordel om man kunne sette maskinen til å gå automatisk videre til å gjennomføre simuleringene.
- **God dokumentasjon.** Så langt har vår erfaring med dokumentasjonen vært god. Det eneste vi har savnet så langt, er dokumentasjon av splintvåpenfila som AVAL genererer. Vi har dessuten fått god hjelp på telefon de gangene vi har hatt behov for det.

8 AVSLUTTENDE KOMMENTARER

Denne rapporten inneholder våre erfaringer med bruken av AVAL så langt. Disse erfaringene har stort sett vært gode. Det er enkelt å gjennomføre simuleringer, og resultatene presenteres på en god måte. Spesielt bra er det at alle filer som AVAL genererer er tekstfiler. Dette forenkler editering en god del. Det er også slik at de fleste fenomener kan beskrives relativt detaljert av brukeren, og dette er også en fordel, selv om vi i noen sammenhenger kunne tenkt oss å legge inn helt egne modeller. Det vil alltid være ting vi hadde ønsket var annerledes, og vi har prøvd å påpeke de svakheter vi har støtt på, og forklart hvorfor vi mener dette er svakheter. Dette er oppsummert i slutten av hvert kapittel der dette er relevant.

Det er svært tidkrevende å generere en målmodell i AVAL. Dette skyldes ikke at AVAL-CAD er et dårlig verktøy, selv om det har forbedringspotensial, spesielt når det gjelder å kunne bruke en eksisterende AutoCAD-modell. Problemet er at det tar lang tid å skaffe til veie den informasjonen som trengs for å generere målmodellen. De data som kanskje er vanskeligst tilgjengelig, er sårbarhetsdata for vitale komponenter. Etter hvert som antall brukere av AVAL øker, håper vi at det utvikles et samarbeid på dette punktet, og at man kanskje kan utarbeide en felles database over sårbarheten til vanlige tekniske komponenter og personell.

Litteratur

(1) FFI, Direktoratet for sivil beredskap og Forsvarsbygg (2002): Forsvarets håndbok i våpenvirkninger.

(2) Logica Limited (1973): Tankkill System, Programmer Guide/User guide.