Angewandte Mathematik und Optimierung Schriftenreihe
Applied Mathematics and Optimization Series
AMOS # 36(2015)

Armin Fügenschuh and Daniel Müllenstedt

Rechnergestützte Konfliktanalyse am Beispiel von Command: Modern Air/Naval Operations

Herausgegeben von der
Professur für Angewandte Mathematik
Professor Dr. rer. nat. Armin Fügenschuh

Helmut-Schmidt-Universität / Universität der Bundeswehr Hamburg
Fachbereich Maschinenbau
Holstenhofweg 85
D-22043 Hamburg

Telefon: +49 (0)40 6541 3540
Fax: +49 (0)40 6541 3672

e-mail: appliedmath@hsu-hh.de
URL: http://www.hsu-hh.de/am

Angewandte Mathematik und Optimierung Schriftenreihe (AMOS), ISSN-Print 2199-1928
Angewandte Mathematik und Optimierung Schriftenreihe (AMOS), ISSN-Internet 2199-1936

Vorwort

Im Frühjahrstrimester (FT) 2015 fand an der Helmut-Schmidt-Universität eine Lehrveranstaltung mit dem Titel “Rechnergestützte Konfliktanalyse” statt. Diese war aufgehängt im Rahmen der Interdisziplinären Studienanteile (ISA). Sie wurde belegt von sieben Studenten aus den Studienfachrichtungen Betriebswirtschaftslehre (5), Volkswirtschaftslehre (1) und Wirtschaftsingenieurwesen (1), welche sich im FT2015 im dritten (6) bzw. sechsten (1) Fachtrimester ihres Studiums befanden.

Die Anregung zu dieser Lehrveranstaltung kam von Mitarbeitern des Planungsamts der Bundeswehr in Ottobrunn. Rechnergestützte Analysewerkzeuge sind dort für verschiedene Fragestellungen bereits im Einsatz. Die Neuentwicklung eines solchen Systems, welche auf die speziellen Anforderungen des Amts zugeschnitten ist, erfordert einen hohen finanziellen Aufwand und einen entsprechenden zeitlichen Vorlauf. Auf der anderen Seite gibt es am Markt erhältliche Programme, welche für vergleichsweise wenig Geld (unter 100 Euro) sofort heruntergeladen und installiert werden können. Ein Beispiel hierfür ist das Programm *Command: Modern Air/Naval Operations* (Version v1.07, Build 678) (kurz: *C:MA/NO*) der Firma Matrix Games, welches dem Bereich des Serious Gaming zugerechnet werden kann.

Serious Gaming bezeichnet eine bestimmte Art von Computerspielen, die sich an der Grenze zur realitätsnahen Simulation bewegen. Die Spieler solcher Spiele legen großen Wert darauf, dass die Realität im Spielgeschehen möglichst detailgetreu nachempfunden wird. Der Detailreichtum bezieht sich dabei nicht zwangsläufig auf die grafische Darstellung (welche bei *C:MA/NO* gemessen an heutigen Standards als eher abstrakt oder primitiv zu bezeichnen ist), sondern auf die Abbildung der agierenden Objekte untereinander in der simulierten Umgebung und mit dem Spieler. Hier zeichnet sich *C:MA/NO* durch die Abbildung einer Vielzahl von militärischen Geräten (Fahrzeugen, Flugzeugen, Schiffen, Sensoren und Aktoren) aus, welche in den Datenbanken des Systems bereits hinterlegt sind und vom Spieler eingesetzt werden können. Die simulierte Welt besteht aus einem Globus, wie man ihn von z.B. GoogleEarth kennt, so dass an jedem beliebigen Ort der Erde der Ablauf eines Konflikts simuliert bzw. durchgespielt werden kann. Beliebig viele Konfliktparteien können mit- oder gegeneinander gegenüberstehen, von denen in der Regel eine vom Spieler gesteuert wird. Eine künstliche Intelligenz (KI) von *C:MA/NO* übernimmt

dann die Rolle der anderen Parteien. Überdies kann sie bei Bedarf den Spieler bei der Steuerung der eigenen Einheiten unterstützen. Aufgrund dieser Rahmendaten wurde das Planungsamt auf *C:MA/NO* aufmerksam.

Ist *C:MA/NO* nur ein detailreiches Spiel, oder lässt es sich darüber hinaus für die Simulation von Konflikten ernsthaft einsetzen? Dieses war die Grundfrage der ISA-Lehrveranstaltung. Die Teilnehmer fanden darauf recht unterschiedliche Antworten. Ausgehend von ihren individuellen Beobachtungen anhand ausgewählter, vom *C:MA/NO*-Hersteller erstellten und mitgelieferten Szenarien arbeiteten sie sich zunächst in die Steuerung des Systems ein und machten erste Gehversuche. In einem zweiten Durchgang erstellten sie eigene Szenarien. Diese waren beispielsweise darauf ausgelegt, die physikalischen und technischen Aspekte eines einzelnen Systems oder einer Komponente eines Systems innerhalb von *C:MA/NO* zu verstehen und mit der Realität zu vergleichen. Andere Teilnehmer erstellten komplexe Szenarien, in denen sich eine Vielzahl eigener und gegnerische Einheiten gegenüber stehen, um die Steuerungs- und Kommandierfähigkeit in *C:MA/NO* zu erproben.

Hamburg,
September 2015

Armin Fügenschuh
Daniel Müllenstedt

Danksagung

Die Lehrenden und die Teilnehmer danken dem Planungsamt der Bundeswehr, Ottobrunn, für die spannende Fragestellung und die gastfreundliche Aufnahme anlässlich der Exkursion im Juni 2015.

Inhaltsverzeichnis

Kapitel 1
Untersuchung des Sonars von U-Booten

Lucas Hüfken

Zusammenfassung Im Folgenden werde ich meine Erfahrungen mit der Simulationssoftware *Command: Modern Air/Naval Operations* beschreiben und dabei auf Situationen eingehen, welche mir positiv sowie negativ aufgefallen sind. Im ersten Teil werde ich mich mit dem Szenario „The Shark“ beschäftigen. Dieses Szenario ist Teil der Software und nicht selbst entwickelt. Anschließend komme ich zu meinem Hauptteil, meiner Untersuchung des Sonars von U-Booten. Dazu haben ich ein eigenes Szenario geschrieben. Abschließend folgt ein Szenario bezüglich Luftbetankung bei *C:MA/NO* sowie ein Fazit, in dem ich wesentliche Erkenntnisse noch einmal darstelle.

1.1 Szenario „The Shark“

Der erste Kontakt zum Simulationsprogramm *C:MA/NO* fand, geschlossen mit der ISA-Gruppe, in einem Computerraum der Universität statt. Wir wurden gemeinsam unter Anleitung an die ersten grundlegenden Tutorials herangeführt. Die Tutorials erklären Schritt für Schritt Grundlagen der Steuerung für Unter-/Überwasser und Lufteinheiten, sowie die Bekämpfung von verschiedenen Zielen. Dabei übernimmt der Computer die Mikrosteuerung der einzelnen Sensoren/Waffensysteme, um den Benutzer zu entlasten. Die ersten Grundlagen sind schnell verinnerlicht, so dass jeder ein eigenes Szenario bekommt, was es gilt ausgiebig zu testen. Mein zu prüfendes Szenario war The Shark. Nach dem Auswählen des Szenarios zeigt das Programm Daten über das gewählte Szenario und erklärt den historischen Kontext sehr umfangreich. Siehe dazu Abb. 1.1.

Nach dem man das Szenario geladen hat, berichtet es über die eigene Truppenstärke und die Stärke der Gegner. Somit bekommt man einen ersten Eindruck,

Lucas Hüfken
Helmut-Schmidt-Universität/Universität der Bundeswehr Hamburg, Jenfelder Allee 70A, 22043 Hamburg, e-mail: w779727@hsu-hh.de

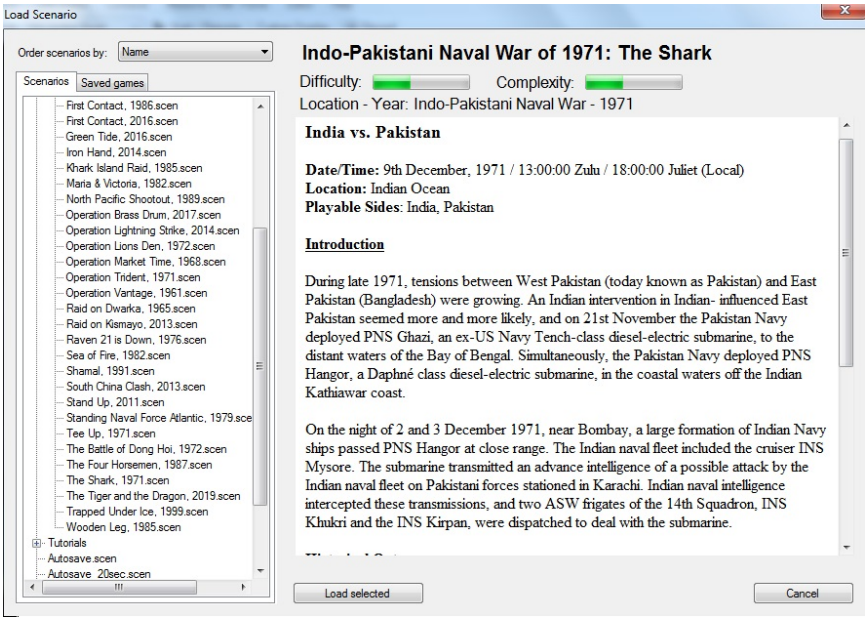

Abbildung 1.1 Missionübersicht und Beschreibung.

was einen erwartet. Außerdem wird der Auftrag, die Mission, die man zu erfüllen hat, genau genannt. In den meisten Szenarien ist nur eine Seite wählbar. In The Shark sind es beide, Pakistan oder Indien. Nach dem Auswählen der Seite (ich wählte für meine Beurteilung des Szenarios die pakistanische Seite), wurde ich direkt ins Geschehen geworfen. Meine einzige Einheit, die PNS Hangor, ein U-Boot der Daphné Klasse, befindet sich in einem Patrouillengebiet vor der indischen Küste. Ich definiere eine ASuW (Anti Surface Warfare) Patrol Mission, wie ich es in den Tutorials gelernt habe und lasse den Computer arbeiten. Mein U-Boot bewegt sich mit Creep (Schleichfahrt) fort und sucht im definierten Bereich nach Überwasserkontakten. Die Zeit des Suchens kann man verkürzen, indem man die Time-Compression von 1-Sec auf z.B. 30-Sec stellt. Damit entspricht eine Sekunde in der Realität 30 Sekunden im Spiel. Diese Funktion ist sehr nützlich, wenn Einheiten eine längere Strecke zum Ziel haben und keine Kontakte zu erwarten sind. Nach kurzer Zeit hat mein U-Boot zwei sich bewegende Sonarkontakte ausgemacht. Alle Kontakte, Waffenwirkungen und sonstige Meldungen werden in einem Chatlog angezeigt. Mein U-Boot steuert nun automatisch mit direktem Kurs auf den nächsten Kontakt zu. Wie ich aus der Missionsbeschreibung am Anfang entnommen habe, erwarten mich als Gegner zwei Fregatten der indischen Marine. Auf der Abb. 1.2 erkennt man die beiden Fregatten.

In den meisten Durchläufen, in denen der Computer die Kontrolle über das Mikromanagement hat, fährt das U-Boot direkt auf den Kontakt zu, feuert eine Zweiersalve Torpedos ab und entfernt sich vom Kontakt. Allerdings ist der Abschusswin-

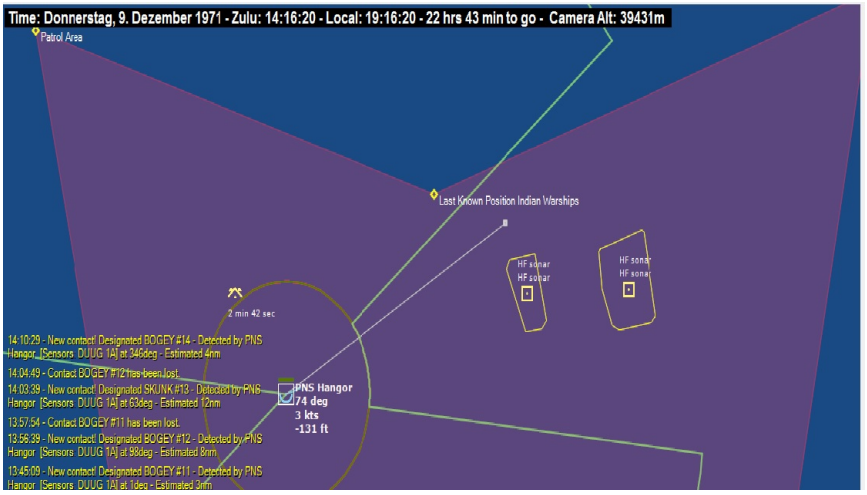

Abbildung 1.2 Die zwei feindlichen Fregatten.

kel sowie die Entfernung zum Kontakt so schlecht gewählt, dass es dem angegriffen Kontakt möglich ist abzudrehen und mit maximaler Geschwindigkeit dem Torpedo zu entkommen. Darauf hin nähert sich das U-Boot wieder dem Kontakt. Wenn das U-Boot zu nah am Kontakt ist, um seine Waffensysteme einzusetzen, entfernt es sich nicht, sondern positioniert sich unter dem Kontakt und dreht sich im Kreis, um Vorauspeilung auf den Kontakt beizubehalten. Dabei wird das U-Boot vom Kontakt erfasst und durch Waffeneinsatz zerstört. Dadurch folgt das Desaster mit einem Final Score von -2500. Dies war kein Einzelfall, sondern ist in den meisten Fällen so aufgetreten, wenn der Computer das Mikromanagement übernommen hat. Eine bessere Möglichkeit für mich, mit einer hohen Erfolgswahrscheinlichkeit, dieses Szenario erfolgreich abzuschließen, bestand darin, dass ich selbst das Mikromanagement übernehme. Dadurch kann der Benutzer selber Kurslinien eingeben und die Geschwindigkeit, sowie die Tauchtiefe variieren. Ich starte die Simulation und lasse mein U-Boot auf die Kontakte mit Creep - Geschwindigkeit, bei 131ft Tauchtiefe, zufahren. Ab ca. 5 NM Entfernung zum Kontakt, wähle ich einen Kurs, um im rechten Winkel auf mein Ziel zuzufahren. Kurz vor meinem Ziel, 0,5 NM, lasse ich zwei Torpedos abfeuern und drehe ab. Die folgenden Treffer versenken die Fregatte. Anhand dieses Szenarios zeigt es sich, dass die Simulation nicht in der Lage ist, günstige Feuerwinkel zu schaffen und bei Gefahr bzw. Nachladezeit sich vom Kontakt zu entfernen. Auch ist die Simulation nicht bemüht, möglichst unentdeckt zu bleiben, da das U-Boot unter dem Kontakt durch taucht und somit entdeckt wird.

1.2 Untersuchung des Sonars

Nach den neuen Erkenntnissen über Command sollten nun selbst Szenarios überlegt und geschaffen werden. Ich wollte das Manöver JTFEX 01-2 aus dem Jahr 2001 nachstellen. Bei dem Manöver waren die Carrier Strike Group der USS Enterprise und das deutsche U-Boot U24 der 206-Klasse beteiligt (siehe Abb. 1.3). Die Aufgabe der U24 war es, unbemerkt den Verteidigungsring der Flugzeugträger-

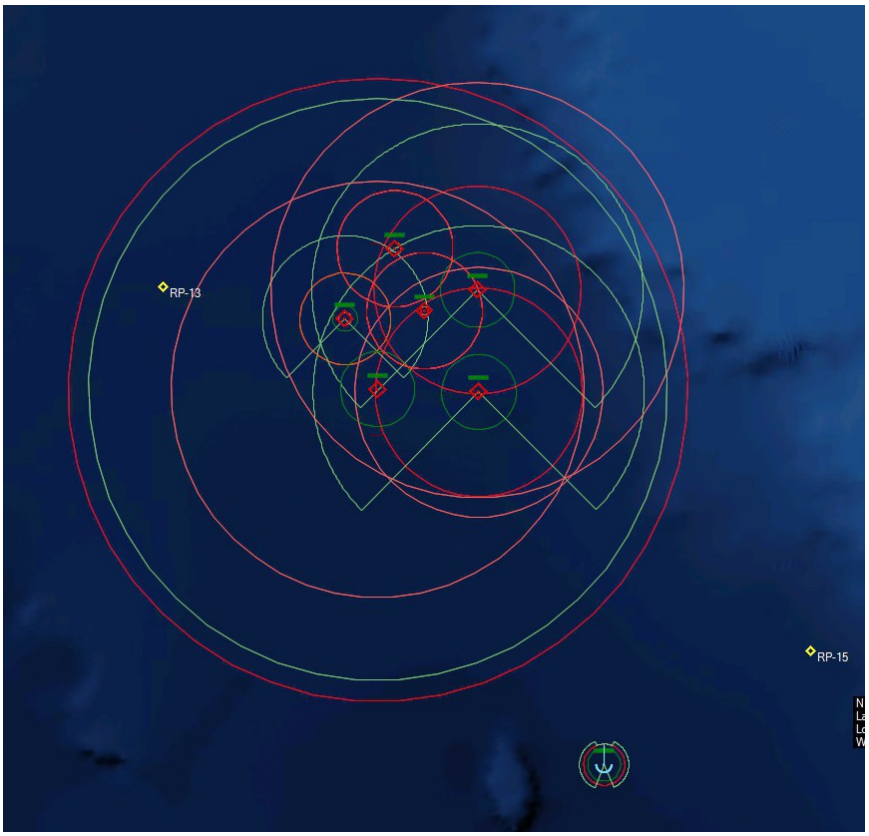

Abbildung 1.3 Die Carrier Strike Group und die U24.

kampfgruppe zu durchbrechen und die USS Enterprise simuliert mit Torpedos zu bekämpfen. Dies gelang der U24 auch und außerdem tauchte sie wenige Meter entfernt vom Flugzeugträger auf, was den darauf stationierten Admiral der US Navy schockierte. Um das Manöver mit dem Simulationsprogramm abzubilden, musste man sich erst einmal mit dem Szenarioeditor vertraut machen. Der Editor war schon nach wenigen Stunden ausprobieren benutzbar. Zuerst müssen Seiten erstellt werden, in diesem Fall USA und Deutschland. Dann kann die Beziehung dieser

beiden Seiten zueinander eingestellt werden. Das Programm bietet eine Datenbank mit einer Vielzahl von Einheiten, die man den erstellten Seiten zuteilen kann. Für mein Manöver stellte ich auf der Seite der USA die Flugzeugträgerkampfgruppe der USS Enterprise und auf der Seite von Deutschland die U24 mit Einheiten aus der Datenbank dar. Für die Flugzeugträgerkampfgruppe definierte ich einen Kurs und wies ihr eine Geschwindigkeit zu, sodass sie größtenteils vom Simulationsprogramm gesteuert wird. Die U24 sollte der Benutzer der Software mit Mikromanagement steuern. Ziel der U24 in der Simulation war es, sich unerkannt der USS Enterprise zu nähern. Nach den ersten Versuchen stellte ich als Benutzer fest, dass es fast unmöglich ist, sich unbemerkt der USS Enterprise zu nähern. Das U-Boot U24 hat frühstens auf 15 NM Kontakte mithilfe des Sonars geortet und so wurde das U-Boot durch die US Zerstörer schnell erfasst und bekämpft. Ich stellte mir die Frage, ab wann U-Boote verschiedene Überwasserkontakte orten und wollte es durch einen Versuch herausfinden. Als Test-U-Boote wählte ich das modernste nicht atomar angetriebene U-Boot der 212-Klasse, das Jagd-U-Boot SSN Seawolf, sowie das älteste U-Boot in der Datenbank die S849 Romeo. Jetzt brauchte ich nur noch Überwasserkontakte, die meine U-Boote mithilfe ihres Sonars erfassen sollten. Dazu wählte ich die Korvette der Independence-Klasse, den Lenkwaffenzerstörer der Arleigh-Burke-Klasse, sowie den Flugzeugträger USS Enterprise. Ich platzierte die Überwassereinheiten ca. 50 NM entfernt von meinem U-Boot und lies sie drauf zu fahren. Diesen Versuch wiederholte ich mit allen drei U-Booten. Außerdem ließ ich die Überwasserkontakte in langsamer (Creep) sowie schneller (Full) Geschwindigkeit fahren. Dabei kam ich auf die, in der Tabelle zu sehenden Ergebnisse. Auch

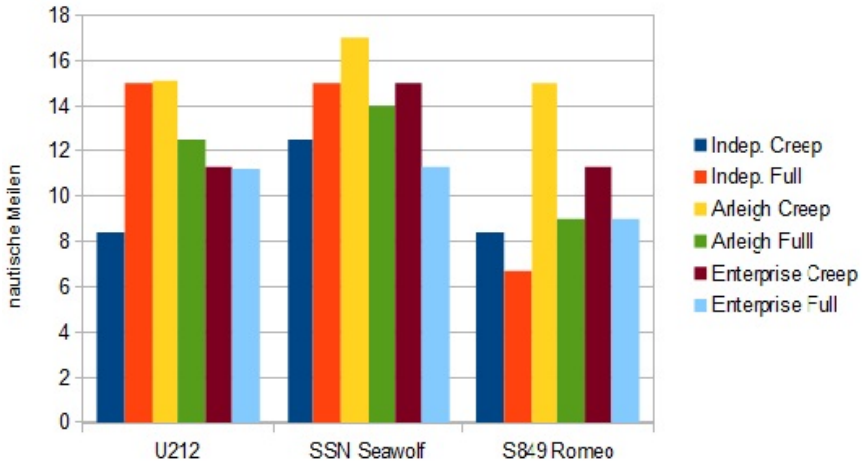

Abbildung 1.4 Entfernung bis zur Aufklärung des Kontaktes in NM.

habe ich festgestellt, dass ein aktives Sonar keinen Unterschied bei der Erfassung macht (siehe Abb. 1.4). Die Wettereinflüsse, in dem Fall Stürme und schlechte See, beeinflussen die Überwassereinheiten nicht. Auch die Tauchtiefe (getestet zwischen 300-66ft) des U-Bootes hat keinen Einfluss auf das Sonar. Interessant ist, dass der Flugzeugträger erst sehr spät erfasst wird, obwohl er doch sehr laut und somit gut hörbar sein sollte. Mir liegen keine realen Vergleichswerte vor, somit kann ich nicht sagen, ob das Sonar im Programm realistisch abgebildet wird.

1.3 Luftbetankung

Ich hatte die Aufgabe, ein Tankflugzeug bzw. seinen Tank zu untersuchen. Die Frage war, ob das Tankflugzeug separate Tanks (für das Flugzeug und für den zu betankenden) hat. Dieses untersuchte ich in einem weiteren selbst erstellten Szenario. Ich erstellte einen Airbus A310 als Tanker sowie einen Eurofighter, welcher in der Luft betankt werden sollte (siehe Abb. 1.5). Sofort wurde anhand der „Fuel“ Anzeige des Airbus klar, dass der Tanker nur einen großen Tank hat, welcher sich schneller leert, sobald der Eurofighter Treibstoff vom Airbus bezieht. Was auffiel, der Eurofighter leerte den Tank des Airbus und brachte diesen somit zum Abstürzen. Auch hier zeigt sich, dass die Simulationssoftware einen Luftbetankungsvorgang darstellen kann, aber die KI nicht in der Lage ist, selbstständig mit dem Betanken aufzuhören, um einen Absturz zu verhindern.

1.4 Fazit

Abschließend kann ich folgendes *Command: Modern Air/Naval Operations* sagen: Der Zeitaufwand, um mit dem Programm vertraut zu werden, ist sehr gering. Auch der Editor ist nach geringer Zeit schon im großen Umfang benutzbar. Die Datenbank bietet viele Einheiten, welche noch durch zusätzliche Ausrüstung verstärkt werden können. Des Weiteren ist die Datenbank beliebig durch Hinzufügen von neuen Einheiten ergänzbar. Kommen wir nun zur Software, wie sie im Spiel funktioniert. Wenn das Programm selbst Kurse oder Angriffswege festlegen soll, beachtet es nicht, ob der Weg optimal ist, sondern wählt immer den direkten Weg, welcher oft fatale Folgen für die Einheit hat. Hier muss der Benutzer viel Mikromanagement beweisen, damit die Mission erfolgreich wird. Dies wird bei Missionen mit vielen Einheiten schnell komplex und unübersichtlich. Auch verfügt die KI nicht über die Fähigkeit zu erkennen, wann sie z.B. Betankungsvorgänge abbrechen muss um, einen Absturz zu verhindern oder vom Ziel abzudrehen, um nicht aufgeklärt zu werden. Im Gesamten sehe ich *C:MA/NO* als Simulationsprogramm für die Bundeswehr geeignet, wenn es darum geht, Gefechte z.B. zwischen Lufteinheiten oder Schiffen zu simulieren. Allerdings muss die Anzahl der beteiligten Einheiten dabei überschaubar bleiben, da der Bediener sonst schnell überfordert ist. Zusammenfas-

Abbildung 1.5 Ein Eurofighter während einer Luftbetankung (Bildquelle: http://images.flugrevue.de/).

send sei gesagt, dass *C:MA/NO* ein Simulationsprogramm mit einer großen Datenbank und vielen Einstellmöglichkeiten im Mikromanagement ist, welches man mit wenig Zeitaufwand gut verwenden kann, um Ausgänge eines bewaffneten Konfliktes zwischen wenigen Einheiten zu simulieren und zu bestimmen.

Kapitel 2
Operation Free Fehmarn

Frederik Scholz

Zusammenfassung Dieser Bericht befasst sich mit den Erfahrungen, der Bedienung und Anwendung der Software *Command: Modern Air/Naval Operations* (kurz: *C:MA/NO*), sowie mit der Szenarioerstellung des manuell erstellten Szenarios *Operation Free Fehmarn*. Insbesondere werden die technische Herangehensweise und die Ausgestaltung dieses Szenarios thematisiert. Der Schwerpunkt liegt hierbei auf der Beantwortung der Leitfrage inwieweit die Software als empirisches Entscheidungs- bzw. Messinstrument für militärische Operationen geeignet ist.

2.1 Erstkontakt mit der Software

Die folgenden Abschnitte beschreiben die ersten Erfahrungen mit der Software *C:MA/NO*. Besonderes Augenmerk liegt hierbei auf der grundsätzlichen Bedienung und der anfänglichen Herangehensweise.

2.1.1 Einstieg

Die Arbeit mit der Software *C:MA/NO* erfordert zunächst eine grundlegende Einarbeitungsphase, bevor mit der eigentlichen Untersuchung begonnen werden kann. Dies liegt unter anderem an der hohen Komplexität der Software. Der Nutzer beginnt anfangs mit der selbstständigen Durchführung von bereits in der Software hinterlegten bzw. vorgefertigten Tutorials (hier: Einstiegsszenarien zur Erklärung der grundlegenden Programmfunktionen).

Frederik Scholz
Helmut-Schmidt-Universität/Universität der Bundeswehr Hamburg, Holstenhofweg 85, 22043 Hamburg, Deutschland, e-mail: f.scholz@hsu-hh.de

Hierbei geht es im ersten von zwei Tutorials um die allgemeine Steuerung von Einheiten, die Zuordnung von Einheiten zu bestimmten vordefinierten Missionen, sowie die erstmalige Bekämpfung von feindlichen Seezielen (in diesem ersten Tutorial werden ausschließlich U-Boote eingesetzt). Hier erhält der Nutzer erste Informationen über die technische Ausstattung von Einheiten (z.B. Sonar) und wie diese einzusetzen ist. Sämtliche Schritte sollten dabei ausführlich von einem erfahrenen Nutzer während des Einstieges erklärt werden, indem dieser beispielsweise an einem separaten Rechner die Mission, sichtbar z.B. über eine Leinwand, parallel mitspielt und die einzelnen Schritte anhand der Szenarienbeschreibung durchführt. Aus eigener Erfahrung ist diese Vorgehensweise wichtig und effektiv, da so erste Hemmschwellen und anfängliche Probleme überwunden werden können.

Im zweiten Tutorial geht es um die Zerstörung feindlicher Schiffe, wobei der Nutzer ebenfalls ein U-Boot steuert. Dieses muss in bestimmte Räume navigiert werden, ohne dabei aufgeklärt zu werden, um anschließend den Kampf gegen feindliche Schiffe zu führen.

Besonders hilfreich zur erfolgreichen Erfüllung der beiden Tutorials sind die jeweiligen Briefings sowie die Erklärungen des erfahrenen Nutzers, da so die einzelnen Schritte zum Erfolg der im Tutorial vorgegebenen Mission nachvollzogen werden können und der Nutzer durch die Einstellungen sowie Abläufe geführt wird.

Im Anschluss an die Tutorials werden die anfänglichen Erfahrungen durch ein erstes, vorgegebenes Szenario vertieft und der Umgang mit der Steuerung von Lufteinheiten erklärt. Hierbei geht es um die Zerstörung feindlicher Bodeneinheiten mit mehreren, dem Nutzer zur Verfügung stehenden Flugzeugen. Das Szenario wirkt durch die vielen zu steuernden Einheiten für einen eher unerfahrenen Nutzer äußerst komplex, daher sollte auch hier, wie die Erfahrung im Kurs „Rechnergestützte Konfliktanalyse“ zeigte, der erfahrene Nutzer die Arbeit begleiten. Ein simples Szenario mit Lufteinheiten kann hier ebenfalls hilfreich sein. Durch das erste, nicht durch Briefings geführte, Szenario erlernt der Nutzer den Umgang mit unterschiedlichsten Flugzeugtypen und den verschiedensten Missionstypen der Software, als auch die Wirkung feindlicher Aufklärung und Waffensysteme.

Eine besondere Schwierigkeit zu Anfang ist vor allem eine fehlende Erklärung aller Menüs und Untermenüs in der Software und welche Funktionen, Einstellungen und Optionen hierdurch angepasst werden können. Dies erschwert zunächst einen schnellen Einstieg in die Arbeit mit *C:MA/NO*. Im weiteren Verlauf der Arbeit wird der Umgang mit den Menüs aber zunehmend einfacher.

2.1.2 Erstes Szenario

Das erste gänzlich eigenständig gespielte Szenario war das Szenario *Battle of Latakia* (siehe Abbildung 2.1). Die Schlacht von Latakia war eine kleine, aber revolutionäre Seeschlacht des Jom-Kippur-Krieges (auch: Ramadan-Krieg oder Oktoberkrieg) vom 6. bis zum 25. Oktober 1973 zwischen Israel und Syrien. Es war die erste Seeschlacht in der Geschichte bei der Raketen und ECM (hier: Electronic Counter

Measures = elektronische Gegenmaßnahmen) zur Kriegsführung eingesetzt wurden. Von der Hafenstadt Latakia kommend, schossen die Syrer ihre Raketen auf die israelischen Schiffe. Israel setzte daraufhin ECM zur Abwehr der Raketen ein, um so die unterlegene Kampfentfernung der eigenen Waffensysteme auszugleichen, und um so die syrischen Torpedoboote auf geringer Kampfentfernung zu bekämpfen.

Festzuhalten ist hierbei, dass das Szenario sehr detailliert, entsprechend der historischen Aufzeichnungen, in der Software nachgestellt worden ist.

Einen ersten Überblick von der Mission, welche Einheiten wie gesteuert bzw. welche Einheiten vernichtet werden müssen, um das Szenario erfolgreich abzuschließen, erhält der Nutzer, indem er es ohne Eingriff ablaufen lässt. Diese Strategie erweist sich auch in anderen Szenarien als sinnvoll, da so bereits im Vorfeld Schwierigkeiten erkannt und Entscheidungen für das weitere Verfahren getroffen werden können. In einem ersten Versuch zur Erfüllung der vorgegebenen Mission erfolgt der erste Eingriff nach feindlichem Beschuss, indem die zu steuernden Einheiten in einen im Szenario vordefinierten Raum navigiert werden, dort eine *Sea Control Patrol* (hier: spezieller Missionstyp in *C:MA/NO* bei dem einer Seeeinheit eine Patrouillenmission in einem bestimmten Raum zugewiesen wird, in welchem es die Kontrolle bzw. Überlegenheit gewinnen soll) gestartet wird und anschließend die feindlichen Einheiten bekämpft werden. Jedoch werden bei dieser Durchführung die neutralen Einheiten, welche sich ebenfalls im Raum befinden, zerstört.

In einem weiteren Versuch werden die Einheiten des Nutzers direkt dem vordefinierten Raum zugewiesen. Dort führen die Einheiten selbstständig den Feuerkampf. Diese Vorgehensweise führt zu der maximal möglichen Punktzahl dieses Szenarios. Auch die im Szenario eingesetzten neutralen Schiffe werden bei diesem Vorgehen nicht wie zuvor im ersten Versuch zerstört.

Das Überleben der neutralen Einheiten hat allerdings keinen Einfluss auf die Punktzahl. Durch die Benutzung des Szenarioeditors lässt sich nach kurzem Durchsuchen der einzelnen Optionsmenüs keine hinterlegte Siegkondition in Zusammenhang mit neutralen Schiffen in diesem Szenario finden.

Bei der Bearbeitung der ersten Szenarien sei anzumerken, dass Voreinstellungen der Szenarien (Missionen, Wegpunkte etc.) überprüft und gegebenenfalls angepasst werden sollten. Auch sollten direkt zu Beginn eigene Befehle erteilt und zunächst der Szenario-Standardverlauf getestet werden. Dies eröffnet die Möglichkeit frühzeitig Handlungsmöglichkeiten zu erkennen.

2.2 Szenarioerstellung

In den folgenden Abschnitten werden die Erfahrungen bei der manuellen Erstellung des Szenarios *Operation Free Fehmarn* beschrieben. Dabei werden die Vorgehensweise und Problemstellungen bei der Erstellung sowie Fehler der Software thematisiert.

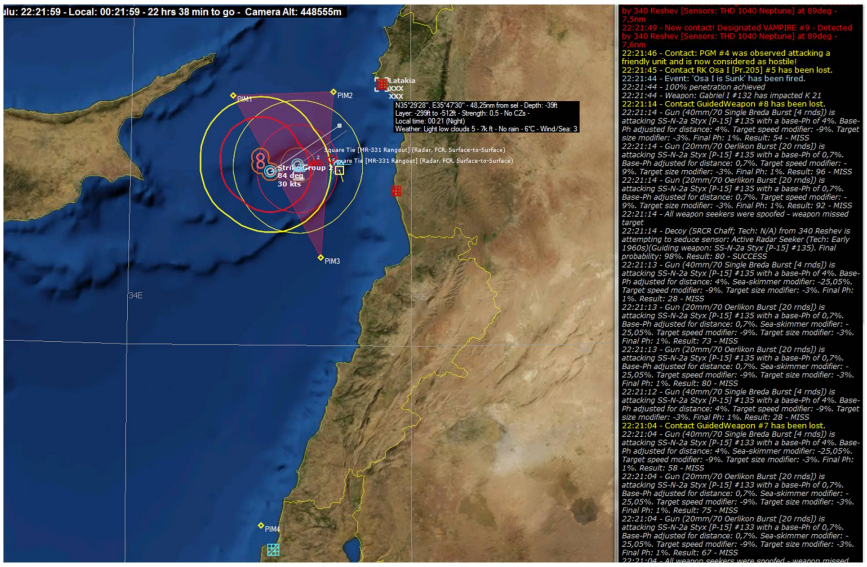

Abbildung 2.1 Ausschnitt aus Szenario *Battle of Latakia*.

2.2.1 Konzept und Idee

Die Programmierung dieses Szenarios dient dazu herauszufinden, ob die Software es ermöglicht Landoperationen zu erstellen, um so die komplexe Kriegsführung zu Land zu simulieren. Da das Programm grundsätzlich für Luft- und Landoperationen ausgelegt ist, wird eine Landoperation auf der deutschen Halbinsel Fehmarn simuliert, um so die in der Software vorprogrammierten Landeinheiten zu testen, inwieweit diese korrekt reagieren und ob generell die Darstellung von asymmetrischer Kriegsführung möglich ist.

Die Halbinsel Fehmarn eignet sich deshalb für eine solche Operation, da bereits mögliche Räume für militärische Landungsoperationen vorhanden sind. Die Anbindung zum deutschen Festland bietet ein strategisch wichtiges Element, das es von den konzipierten Aufständischen zu halten gilt. So stellt eine Landungsoperation von Spezialkräften ein realistisches Szenario dar.

2.2.2 Umsetzung

Vor der eigentlichen Szenarioerstellung werden die benötigten Einheiten in der Datenbank der Software ausgewählt. Diese patrouillieren anschließend probeweise auf Fehmarn und kämpfen gegeneinander, nachdem zuvor die entsprechenden Parteien des Szenarios definiert werden. Die Parteien in diesem rein fiktiven Szenario beste-

hen aus der von dem Spieler spielbaren Partei der Bundeswehr, den Aufständischen der FFF (hier: die Fraktion „Fehmarnsche Freiheitsfront“) und der Zivilbevölkerung.

Die eingesetzte Infanterie arbeitet zunächst reibungslos, stellt aber nach kurzer Zeit sämtliche Aktivitäten ein. Das Szenario läuft währenddessen weiter, ohne das etwas geschieht. Nach mehrmaligen Tests stellt sich heraus, dass die Landeinheiten, nachdem sie ihre Munition verschossen haben, einfach stehenbleiben. Daher werden die Einheiten mit mehr Munition ausgestattet. Wie dies funktioniert, lässt sich durch eine Recherche in Internetcommunities herausfinden, die dankenswerter Weise sehr aktuelle und detaillierte Beschreibungen beinhalten. Diese große, aktive Community hilft auch später bei der Programmierung von Skriptbefehlen mit dem in der Software einsetzbaren LUA-Script (hier: erweiterbare Skriptsprache, welche Programmierbefehle interpretieren kann u.a. für die Programmierung von Spielen) weiter, z.B. zur Teleportation von Einheiten.

Nachdem die eingesetzten Landeinheiten die beabsichtigten Aktionen ausführen, werden erste Gebäude über den Szenarioeditor hinzugefügt, welche der FFF zugeordnet werden. Im Anschluss soll eine Brücke zwischen Festland und Halbinsel erstellt werden, was mit der Software bis zu diesem Zeitpunkt allerdings nicht möglich ist. Eine Brücke wird in der Software als ein einzelnes Gebäude ohne Anfang und Ende dargestellt, also lediglich als eine Fläche. Da dies der Absicht einer detaillierten und möglichst realistischen Darstellung einer Landoperation widerspricht, wird lediglich eine feindliche Basis in der Stadt Burg auf Fehmarn erstellt. Dort werden sowohl Wohnhäuser der Zivilbevölkerung, als auch feindlich besetzte Wohngebäude implementiert.

Durch die fehlende Brücke fehlt das zuvor angesprochene strategisch wichtige Element. Daher wird ab dieser Stelle die Idee des Transportes vom Spieler steuerbaren Infanterieeinheiten über den Seeweg fokussiert und so lediglich der Kampf auf der Halbinsel geführt. Statt Kampffahrzeugen wird nun ein geeignetes Marineschiff eingesetzt bei dem es herauszufinden gilt, wie es möglich ist, die Einheiten von dem Schiff an Land zu bringen. In den vorgegebenen Programmieroptionen der Software muss daher nach der entsprechenden Funktion gesucht werden, welche allerdings, aufgrund unzureichender Funktionen innerhalb der Software, nicht existiert. Einheiten können allerdings auch durch eine Verknüpfung sogenannter *Events* (hier: aufeinanderfolgende oder einzelne, unabhängige Ereignisse als Auslöser von bestimmten Programmabläufen) in der Software teleportiert werden. Mithilfe dieser Funktion werden die Infanterieeinheiten von einem Punkt der Weltkarte in die gewählte Landungszone teleportiert, sobald das Schiff den durch Wegpunkte vordefinierten Raum erreicht. Selbiges wird auch für die Gebäude und Infanterieeinheiten der FFF sowie der Zivilbevölkerung programmiert.

Diese Lösung ist jedoch aufgrund der Realitätsferne eher suboptimal, da es die Teleportation von Einheiten von einem Punkt zu einem anderen von Anfang an zu vermeiden gilt. Stattdessen muss ein vorprogrammierter Auslöser gefunden werden. In Internetforen stößt man auf der Suche nach einer geeigneten Lösung schnell auf das sogenannte LUA-Script, mit dessen Hilfe auch komplexere Abfolgen von Ereignissen und Befehlen im Editor programmiert werden können. Nach einigen

Stunden des schlichten Ausprobierens gelingt es letztlich, die Einheiten durch einen programmierten Befehl in das Szenario automatisch einzusetzen, ohne diese vorher platzieren oder teleportieren zu müssen.

Jedoch kann so den eingesetzten Einheiten nur die vordefinierte Bewaffnung zugewiesen werden, da verschiedene Bewaffnungsoptionen dieser Art Einheiten nicht vorgesehen sind und in der Datenbank der Software selbst geändert werden müssen. Dieser Schritt ist jedoch für die Erstellung des Szenarios und die aufgetragene Aufgabe zu umfangreich und daher zu vernachlässigen. Allerdings ist dies laut Foren und Communities definitiv möglich.

Das Szenario ist zu diesem Zeitpunkt zwar spielbar, allerdings fehlen noch die Siegkonditionen, sodass das Szenario endlos weiter laufen würde, obwohl sämtliche feindlichen Einheiten bekämpft werden. Weiterhin sollen Fußgänger in das Szenario integriert werden, da es für einen Spieler zu einfach ist, die feindliche Infanterie zu identifizieren. Auch Seeminen sollen eingesetzt werden, die zuerst vernichtet bzw. aufgeklärt werden müssen, bevor die Landungsoperationen beginnen kann.

Zunächst müssen geeignete Seeminen recherchiert und so eingesetzt werden, dass ein gefahrloses Durchfahren des Gewässers bis hin zur Landungszone unmöglich ist. Um dieses Ziel zu erreichen, muss das vermutete Minenfeld zunächst von einem Minenräumschiff durchfahren werden. Auch dieses wird mit einer vordefinierten Minenräummission im Editor eingesetzt, sodass der Spieler lediglich den Befehl zur Räumung geben und über den Einsatz von Sensoren zur Minenaufklärung entscheiden muss.

Als Fußgänger werden unbewaffnete Infanterieeinheiten auf der Seite der Zivilbevölkerung hinzugefügt. Diese patrouillieren mit den Einheiten der FFF im gleichen Raum, da der Spieler so die feindlichen Einheiten nicht direkt ausmachen kann. In einem ersten Test werden die Fußgänger jedoch mit dem taktischen Symbol für *bekämpft* (hier: eine ausgefallene Entität in einem Gefecht, üblicherweise gekennzeichnet durch ein mit einem Kreuz durchgestrichenes taktisches Symbol) dargestellt. Daher müssen die Fußgänger ebenfalls mit Waffen ausgestattet werden, sodass diese ein realistisches Verhalten an den Tag legen, jedoch durch entsprechende Voreinstellungen nicht kämpfen können. Dies erfolgt über sogenannte „Doktrineinstellungen“.

Die Siegkonditionen werden mithilfe des Editors festgelegt. Dabei sind zwischen dem Spieler gut geschriebenen Punkten für erfolgreich bekämpfte feindliche Einheiten und Minuspunkten für den Verlust eigener Einheiten und die unbeabsichtigte Bekämpfung von unbeteiligten Zivilisten zu unterscheiden. Das Räumen des Minenfeldes oder einzelner Minen kann allerdings nicht als Erfolg programmiert werden, da dies mit der Software zu diesem Zeitpunkt nicht möglich ist. Jedoch ist dies als positiv und realistisch zu bewerten, da ein Minenfeld, sobald es einmal gelegt wurde, nie mit hundertprozentiger Sicherheit geräumt werden kann. Daher ist es hier dem Spieler überlassen zu entscheiden, wann er das Minenfeld für geräumt erklärt und mit dem Transportschiff in Richtung Landungszone aufbricht.

Da es sich bei diesem Szenario um einen sehr kleinen Ausschnitt der Weltkarte handelt, kann mit Hilfe eines zusätzlichen Programms und der Software *C:MA/NO* ein Overlay von Fehmarn hinzugefügt werden. So sind viele Details der Halbin-

sel erkennbar, welche allerdings nur einen optischen Effekt für den Spieler, jedoch keinerlei Auswirkung auf die Software oder den Ablauf des Szenarios haben. Die Anleitung zur Einbindung eines Overlay ist dem Usermanual zu entnehmen.

Zusätzlich soll den vom Spieler steuerbaren Infanterieeinheiten eine Minidrohne zur Verfügung gestellt werden, um so die eindeutige Aufklärung der feindlichen Infanterie und Gebäude zu gewährleisten, da durch die Namensgebung der beiden Parteien FFF und Zivilbevölkerung (beide Parteien heißen im Szenario „Zivilbevölkerung“) eine Aufklärung der jeweiligen Zugehörigkeit der einzelnen Einheit nicht möglich ist. Jedoch sind die feindlichen Einheiten so passiv eingestellt, dass diese erst nach einem Beschuss des Spielers ihre wahre Identität bzw. Zugehörigkeit offenlegen, und so durch keine in der Software verfügbare kleine oder große Drohne eindeutig aufgeklärt werden können. Diese Erkenntnis erlangt man beispielsweise durch den Einsatz verschiedener, verfügbarer Drohnentypen in der Operation. Die Drohnen klären den Bereich wie gewollt auf, können allerdings nur neutrale Einheiten identifizieren. Eine feindliche Zuordnung bzw. eine eindeutige Identifikation feindlicher Einheiten erfolgt dementsprechend nicht.

Das so entstandene Szenario *Operation Free Fehmarn* (siehe Abbildung 2.2) steht zum Download zur Verfügung.

Abbildung 2.2 Ausschnitt aus Szenario *Operation Free Fehmarn.*

2.3 Softwarebetrachtung

Im Folgenden werden die allgemeine Handhabung und die Bedienung der Software *C:MA/NO* analysiert. Weiterhin werden beobachtete Softwarefehler beschrieben.

2.3.1 Allgemeine Handhabung

Die Arbeit mit der Software erweist sich anfangs als schwierig. Das Usermanual hilft dem Nutzer zwar grundlegende Funktionen zu verstehen, allerdings hilft dies lediglich während der Arbeit mit der Software weiter. Ein schlichtes Einlesen zu Beginn der Arbeit mit der Software erweist sich als zu komplex, um sämtliche Funktionsweisen und Zusammenhänge verstehen zu können. Gerade die Durchführung erster Missionen, die Einheitensteuerung und die Ausführung einzelner Befehle sind nach der achtwöchigen Erfahrung mit der Software als kompliziert und langwierig zu bewerten. Die Einweisung eines erfahrenen Nutzers ist essenziell wichtig.

Nach Absolvieren der Tutorials erhält der Nutzer eine grundlegende Idee der Funktionsweise der Einheitensteuerung, jedoch helfen zur Lösung anderer Szenarien teils nur die Briefings der Missionen und Erklärungen im Usermanual weiter. Bei extrem komplexen und besonders schwierigen Szenarien helfen ggf. nur Videos von anderen Nutzern auf Videostreamingplattformen weiter oder die gemeinsame Diskussion im Kreis anderer Nutzer, in diesem Fall die Kursteilnehmer.

Die Bedienung der Software ist nur mäßig intuitiv, da Nutzer durch viele Untermenüs klicken müssen, die teils nur wenig bis gar nicht erklärt werden.

In diesem Zusammenhang ist auch zu erwähnen, dass Szenarien nicht auf einen bestimmten Zeitpunkt zurückgesetzt werden können, falls Einstellungen und Entscheidungen fälschlicherweise gewählt oder getroffen wurden. Hierzu muss das Szenario neu gestartet werden, wobei alle zuvor gewählten Einstellungen verfallen. Ein Problem hierbei ist auch, dass Szenarien nicht nach einer festen Handlung verlaufen und so eine Wiederholung unter gleichen Einstellungen nicht möglich ist. Dies liegt insbesondere an zufällig definierten, voreingestellten Patrouillerouten von Einheiten und an unterschiedlichen Wirkungen von Waffensystemen nach einer bestimmten Zufallswahrscheinlichkeit.

Die Erstellung von komplexen Szenarien ist sehr kompliziert und erfordert Vorkenntnisse mit, dem in der Software nutzbaren, LUA-Script. Nur so lassen sich erweiterte Vorhaben, wie etwa die Teleportation von Einheiten und / oder Auslöser in der Erstellung von Szenarien programmieren und realisieren. Um also die Software problemlos nutzen und bedienen zu können, ist eine längere Einarbeitungsphase unabdinglich.

2.3.2 Beobachtete Fehler

Während der Arbeit mit der Software, beim Durchlaufen oder bei der Erstellung von Szenarien, sind folgende Fehler oder Probleme aufgetreten:

2.3.2.1 Fehlerhafte Routenberechnung im Szenario *Wooden Leg*, 1985

Die Einheit *340 Aliyah [Saar 4.5, Helo Pad]* berechnete keine automatische Route über den Seeweg, nachdem ein Angriffsbefehl erteilt wurde. In diesem Szenario wählte die KI (hier: künstliche Intelligenz) der Software für das Helo Pad die kürzeste, direkte Route zum Ziel. Dadurch fuhr es in Richtung Küste und drehte sich dort auf der Stelle, da das Ziel nicht in Reichweite der Waffensysteme lag und die Route durch Landmasse blockiert war. Um das Ziel bekämpfen zu können, musste eine manuelle Route um die Landmasse herum vorgegeben werden, erst dann bekämpfte die Einheit das Ziel ordnungsgemäß (siehe Abbildung 2.3).

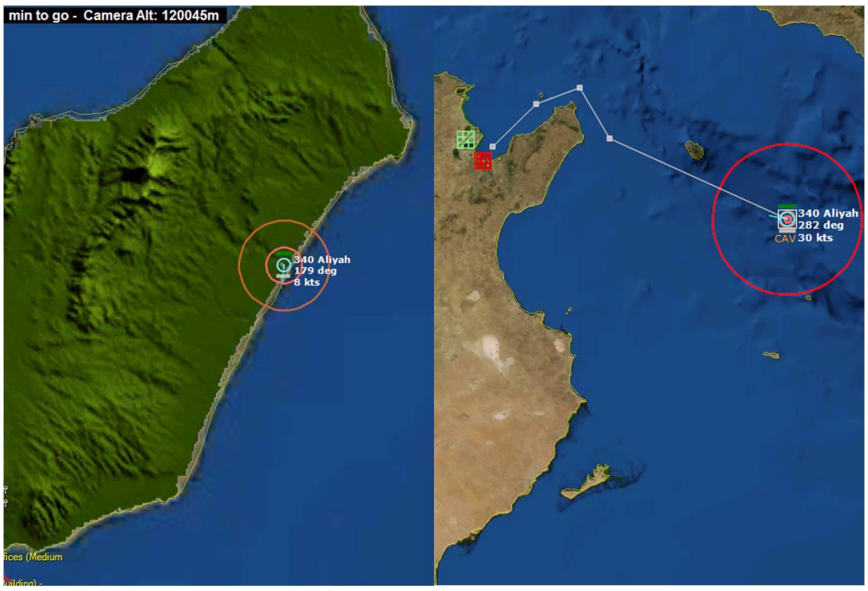

Abbildung 2.3 Ausschnitte aus Szenario *Wooden Leg*.

2.3.2.2 Fehlerhafte Einstellung bei verschossener Munition

Wie oben bereits erwähnt, bewegten sich oder kämpften Landeinheiten nicht mehr, sobald sie ihre Munition verschossen hatten. Sie ließen sich dann auch durch manuelle Routenzuweisung nicht mehr steuern. Dies war sowohl bei vom Spieler steuerbaren Einheiten, als auch von der KI gesteuerten Einheiten zu beobachten.

2.3.2.3 Unzureichende Programmierung von Gebäuden

Während der Erstellung von Szenarien fiel auf, dass Infanterieeinheiten, welche sich hinter Gebäuden befanden, durch direkte Waffenwirkung u.a. von Maschinengewehren getroffen werden konnten. Ein Test mit einer von vielen Gebäuden umringten Infanterieeinheit und direktem Beschuss durch feindliche Infanterieeinheiten zeigte, dass Gebäude lediglich als ebene Fläche von der Software dargestellt wurden (siehe Abbildung 2.4). Obwohl entsprechende Höhenmaße in der Datenbank vorhanden waren, wurde die Infanterieeinheit in der Mitte des Gebäudekreises problemlos getroffen; Gebäude hielten dabei die Geschosse nicht ab.

Abbildung 2.4 Beschusstest einer von Gebäuden umgebenen Einheit.

2.4 Beantwortung der Leitfrage

Zusammenfassend ist zu sagen, dass die Software *C:MA/NO* als empirisches Entscheidungs- bzw. Messinstrument in Zusammenhang mit Landoperation nicht geeignet ist. Die unzureichende Programmierung von Landeinheiten ist bei dieser Bewertung maßgeblich. Vor allem Wahrscheinlichkeiten bei Waffenwirkungen und immer wieder neu einzugebende Einstellungen zu Beginn einer neuen Untersuchung, und damit einhergehende Messungen und Tests, erschweren eine kontinuierliche Wiederholung ein und desselben Versuchsaufbaues. Eine Messung von Ergebnissen zu einem Versuch sind dadurch eher nicht möglich.

Jedoch muss an dieser Stelle auch betont werden, dass das Programm primär für See- und Luftoperationen ausgelegt ist. Erfahrungen im Kurs haben gezeigt, dass es durchaus Einsatzmöglichkeiten für die Software *C:MA/NO* als Entscheidungsinstrument gibt. Grundsätzlich bedarf die Arbeit mit *C:MA/NO* aber einer generellen Einweisungsphase, um so alle Funktionsweisen kennenzulernen.

Die Prüfung von tatsächlichen Einsatzmöglichkeiten ist allerdings nicht Bestandteil dieses Berichtes.

Kapitel 3
Angriff auf Rebellen in einer Wüstenregion

Alexander Dirschke

Zusammenfassung Ziel dieser Arbeit ist es, aufgrund eigener Erfahrungen mit dem Spiel *Command: Modern Air/Naval Operations* eine Aussage über dessen Realitätsnähe zu treffen. Es gilt daher zu überprüfen, inwiefern *C:MA/NO* nur als Spiel oder auch als realistische Simulation zu verstehen ist. Dies geschieht anhand zweier Szenarien, die ich mithilfe der Software durchgespielt habe. Eins davon ist ein vom Spielehersteller entwickeltes und mitgeliefertes Szenario, während das andere von mir entworfen wurde. Es werden taktische und strategische Hintergründe beleuchtet, sowie Waffensysteme und deren Wirkung auf Plausibilität hin untersucht.

3.1 Erste Erfahrungen

Anhand vorgegebener Übungsszenarien konnte ich einen ersten Eindruck von dem Spiel gewinnen. Es gibt drei Übungen, in denen man sämtliche Verfahren kennenlernt, um Einheiten auf der Karte zu bewegen und Operationen mit ihnen auszuführen. Zwei der Übungen basieren auf Seeoperationen und die dritte wird in der Luft ausgeführt.

In der *ersten Übung* ist es die Aufgabe des Spielers, ein U-Boot zu befehligen und ihm Kurse und Aufgaben zuzuweisen. Zunächst geht es darum, einen Patrouillenweg entlang zu fahren, in dessen Verlauf man ein feindliches U-Boot erkennt und es bekämpfen muss. Später gilt es, vom Wasser aus gegen Luft- und Landziele zu wirken.

Die *zweite Übung* ist der ersten sehr ähnlich, wobei man hier wiederum das Kommando über ein Schiff besitzt. Auch hier müssen Landziele und nachrückende feindliche Einheiten bekämpft und Angriffe abgewehrt werden.

Alexander Dirschke
Helmut-Schmidt-Universität/Universität der Bundeswehr Hamburg, Holstenhofweg 85, 22043 Hamburg, e-mail: w779681@hsu-hh.de

Die *letzte Übung* ist allerdings komplizierter, da hier eine feindliche Basis aus der Luft ausgeschaltet werden muss. Eine Luftabwehr des Gegners in Form von Raketen und Abfangjägern erschweren dieses Vorhaben zusätzlich. Man muss im Vorfeld viele Überlegungen anstellen, wie man gegen den Feind vorgeht, um die eigenen Verluste möglichst zu minimieren. Es stellt sich schließlich als nützlich heraus, zuerst die feindlichen Radarstellungen zu zerstören. Dazu muss man die Flughöhe der Bomber anpassen und gegebenenfalls eine taktische Luftbetankung einsetzen.

An diesem Beispiel zeigt sich das enorme Potential des Spiels, da es unzählige Variablen und Einstellungen gibt. Alle Veränderungen haben letztlich einen Einfluss auf den Erfolg und den Ausgang der Mission.

Insbesondere Luftoperationen sind bei *C:MA/NO* sehr komplex abgebildet. Diese Tatsache war für mich der Anlass, mich später in meinem eigenen Szenario mit dieser Art der Kriegsführung näher zu beschäftigen.

Dabei waren die Erfahrungen und Erkenntnisse wertvoll, die ich aus dem „Khark Island Raid“ gewinnen konnte. Dieses Szenario ist im nächsten Kapitel näher beschrieben.

3.2 Khark Island Raid

Der *Khark Island Raid* ist eine Luftoperation ausgehend vom Irak und findet im Jahr 1985 statt. Ziel dieser Mission ist die Zerstörung iranischer Ölanlagen auf der Insel Khark im Persischen Golf. Die Luftschläge sind historisch belegt, allerdings ist die Anzahl der Angreifer und die Art ihrer Bewaffnung fiktiv.

Der Angriff erfolgt durch *sechs SU-22 Fitter* und *sechs Mig 23 Flogger*, wobei die *Fitter* den tatsächlichen Angriff ausführen und die *Flogger* den Begleitschutz darstellen. Ausgerüstet sind sie mit AA-7 Apex C Raketen, ungelenkten Bomben, sowie mit Chaffs und Flares zum Selbstschutz. Diese Bewaffnung war in der Mission vorgegeben und konnte von mir nicht nachträglich verändert werden.

Die feindlichen Kräfte verfügen über eine unbekannte Anzahl an *F-4 Phantom* und *F-5 Tiger*, sowie über Flugabwehrraketen auf der Insel selbst.

Anfangs habe ich versucht den Flugzeugen einer reguläre Angriffsmission zuzuweisen. Allerdings wurden meine Kräfte nach dem Aufstieg vom feindlichen Radar erkannt und von Jägern abgefangen. Es wurde daher schnell klar, dass ich über das sogenannte Mikromanagement meine Flugzeuge im Tiefflug koordinieren musste. Mikromanagement bedeutete für mich wiederum, aktiv in das Spiel einzugreifen und die Einheiten über konkrete Anweisungen direkt zu befehligen. Somit konnte ich die künstliche Intelligenz meiner Truppen umgehen, um den Angriff nach meinen Wünschen durchzuführen. Ziel war es, mit den *Fittern* fast auf Meereshöhe (Emcon settings: Radar off) an die Insel heran zu fliegen, um anschließend kurz vor Erreichen der Insel auf 500m zu steigen und die Bomben ins Ziel zu bringen. Dies war notwendig, um die Mindestabwurfhöhe der Bomben einzuhalten und um vom feindlichen Radar erst spät erkannt zu werden. Trotzdem konnten die Bomber durch

die stationierten HAWK Raketen bekämpft werden, noch ehe sie in die Wirkungsreichweite ihrer Waffen gelangten. Dies ist in Abbildung 3.1 zu erkennen.

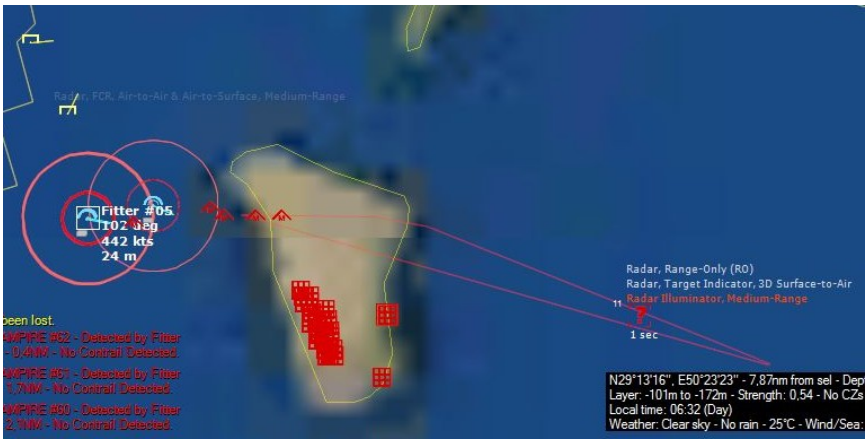

Abbildung 3.1 Angriff auf die Insel und anfliegende Raketen.

Die Bomber haben zwar versucht, Abwehrmaßnahmen einzuleiten, sind allerdings in der Mehrzahl gescheitert und daher abgeschossen worden. Hier konnte ich erfahren, dass die Radarstrahlen sich tatsächlich, wie in der Realität auch, optisch ausbreiten.

Ich habe ebenfalls versucht mich, der Insel zu nähern, um dann nach einer feindlichen Radarerfassung mit Nachbrenner von der Insel weg zu fliegen. Sobald der Gegner seine Raketen in meine Richtung verschossen hatte, plante ich anschließend den Angriff. Dieses Vorhaben scheiterte jedoch ebenso, da der Gegner schlicht über zu viele Raketen verfügte und herbeigerufene Abfangjäger meine Flugzeuge abschossen. Hier zeigte sich auch wiederum die Unterlegenheit meines Begleitschutzes im Luftkampf.

Trotz unzähliger Versuche konnte ich meine Mission nicht erfolgreich abschließen. Auch der spielinterne Zufallsgenerator, der beispielsweise die Patrouillenrouten des Gegners berechnet, konnte das Ergebnis nur geringfügig zu meinen Gunsten beeinflussen. Die Tatsache, dass meine Kräfte schlecht ausgerüstet waren und die Feindlage in dieser Mission völlig unbekannt ist, haben das Szenario in meinen Augen unlösbar gemacht.

Die Daten über die Bewaffnung und die Flugzeuge sind tatsächlich korrekt, was das Spiel aufwertet. Auch die Schadensmodelle sind realistisch und nachvollziehbar. Außerdem gibt extrem viele Möglichkeiten, die Flugzeuge zu befehligen. Der größte Nachteil ist das Mikromanagement. Bei einer Anzahl von zwölf Luftfahrzeugen ist es schwer, alle einzeln zu bewegen. Weist man beispielsweise die Flugzeuge an, den Angriff in der Mindesthöhe zu fliegen, so tun sie dies, ohne die Mindestabwurfhöhe der Bomben zu berücksichtigen. Dieses verhindert letztlich einen erfolgreichen Angriff, da die Maschinen durch den vom Spieler auferlegten Befehl ihr Ziel

nun nur noch mit der Bordkanone bekämpfen können. (Die Mindestflughöhe der Bomber lag hier bei 60-100m und die Abwurfhöhe bei etwa 500m über dem Grund). Dies hat zur Folge, dass Flugroute, Geschwindigkeit und Flughöhe der Flugzeuge permanent angepasst werden müssen. Durch diese äußerst komplexe Steuerung der Einheiten leidet wiederum die Spieldynamik. So entsteht der Eindruck, der Spieler müsse seine Truppen programmieren. Die Erfahrungen, die ich in dieser Mission gemacht habe, wollte ich nun für mein eigenes Szenario nutzen. Dieses wird im folgenden Kapitel behandelt.

3.3 Luftangriff in einer Wüstenregion

Das folgende Szenario wurde von mir mit dem Editor erstellt. Meine Absicht war es, eine Krisensituation in einem realitätsnahen Konfliktgebiet durch *C:MA/NO* abzubilden.

Dabei war es mir wichtig, moderne Waffensysteme zu nutzen und zu simulieren, ob eine asymmetrische Kriegsführung mit dem Spiel darzustellen ist. Die Mission findet in einer trocken-heißen Wüstenregion statt. Terroristische Gruppierungen führen zu einer politischen Destabilisierung im Norden des Landes und müssen bekämpft werden. Dabei kommen abwechselnd Drohnen und Erdkampfflugzeuge zum Einsatz.

Ich bin hierbei davon ausgegangen, dass die Feindlage unbekannt ist und die gegnerischen Truppen über keine Luftkräfte verfügen. Es ist allerdings mit einer feindlichen Luftabwehr zu rechnen. Auf der Seite der Regierung stehen insgesamt *vier F-16 Desert Falcon*, *vier Tornado IDS* und *eine Predator* Aufklärungsdrohne zur Verfügung. Die Kampfflugzeuge sind hierbei für den Erdkampf ausgerüstet. Die *Predator* wird in diesem Szenario sowohl zur Aufklärung, als auch für den Bodenangriff genutzt. Für diese Rolle erhält sie neben ihrer vorhandenen Sensorik eine Bordkanone. Neben der *Predator* existieren übrigens viele weitere spielbare Drohnen, wie die *Global Hawk* oder die *Heron*. Die Jets sind im Süden des Landes stationiert, während die Drohne von Anfang an aus der Luft operiert.

Die feindlichen Kräfte sind mit drei regulären Einheiten dargestellt: Mörser, Infanterie und Luftabwehr. (Die Luftabwehr wird mit Stingerraketen ausgerüstet). Außerdem werden im weiteren Verlauf diverse zivile Gebäude als besetzt simuliert. (Die Gebäude werden mit Gewehren „ausgerüstet“). Ich habe schließlich für mein Szenario zwei Angriffsmöglichkeiten durchgespielt.

1. Aufklärung und Angriff mit der Drohne
2. Aufklärung durch die Drohne und anschließender Einsatz von Kampfjets zur Bodenzielbekämpfung

Für den ersten Durchlauf habe ich für die Drohne eine *Patrolmission* erstellt. Das Patrouillengebiet (im Zentrum des Landes) beschränkt sich dabei auf die Fläche, in der die feindlichen Bodentruppen vermutet werden.

Die *Predator* überfliegt den Raum und erkennt zuverlässig die gegnerischen Einheiten. Da das Verhältnis der beiden Parteien (Regierung und Terroristen) zueinander auf feindlich eingestellt wurde, bekämpft die Drohne die Einheiten unmittelbar nach deren Aufklärung. Die Luftabwehr ist gegenüber der Drohne wirkungslos, da sie außerhalb der Reichweite der Stingerraketen operiert. Die maximale Bekämpfungshöhe beträgt zwischen 3000 und 3500m. Auch die als feindlich besetzt simulierten Gebäude werden von der Drohne zwischen den zivilen und freundlich gesinnten Häusern erkannt.

Hier wählt die Drohne wiederum eine niedrigere Flughöhe (etwa 1700m), um mit der Kanone zu wirken. Die Gebäude erwidern das Feuer zwar, konnten die Drohne aber in keinem Durchlauf abschießen. Hier spielte vermutlich die Größe der Drohne die entscheidende Rolle.

Für den zweiten Durchlauf habe ich die unbemannte *Predator* für die Aufklärung genutzt und habe die *Tornados* und *F-16* für den Angriff ausgewählt. Dabei habe ich versucht, die Nähe von zivilen und feindlichen Gebäuden zu variieren.

Zunächst überfliegt die Drohne das ihr zugewiesene Patrouillengebiet, um die Bodentruppen zu lokalisieren. Anschließend fliegen die Jets in relativ niedriger Höhe an und bekämpfen die ausgemachten Ziele erwartungsgemäß. Hier kommt es jedoch erstmals zu Verlusten, da die Flugabwehrraketen die Bomber erfassen können und auf sie abgeschossen werden. Diese verteidigen sich genau wie im *Khark Island* Szenario oder weichen aus. Bei dem Angriff auf die Gebäude verhält es sich ähnlich. Auch hier werden die Flugzeuge be- beziehungsweise in seltenen Fällen sogar abgeschossen.

Insgesamt lässt mein Szenario einige Stärken der Software erkennen. Moderne Waffensysteme sind genauso zu finden wie alte. Technische Daten wie etwa Größe, Reichweite oder maximale Flughöhe sind gut recherchiert und technisch korrekt. Ausrüstungsmöglichkeiten und Spezifikationen sind realistisch, abgesehen davon, dass hier eine Drohne mit einer Panzerhaubitze oder Ähnlichem ausgerüstet werden kann. Das Verhalten der Flugzeuge ist nachvollziehbar und wie auch im ersten Szenario bis ins Detail vom Spieler beeinflussbar.

Allerdings zeigen sich auch diverse Schwächen des Spiels. Es kam beispielsweise zu widersprüchlichen Ergebnissen bezüglich des Trefferbildes der Flugzeuge. So habe ich ein feindliches Gebäude unmittelbar neben einem zivilen platziert. Wird nun eine 1000 Pfund Bombe auf das Ziel abgeworfen, so wird lediglich das feindliche Haus zerstört. Die zivile Einrichtung bleibt hingegen unversehrt. Auch wenn z.B. Hochhäuser scheinbar aufeinander gebaut werden, so bleibt das Ergebnis dasselbe. Daraus habe ich den Schluss gezogen, dass sämtliche Gebäude in *C:MA/NO* nicht dreidimensional abgebildet werden können. Darüber hinaus scheint eine Streuwirkung von Munition im Spiel nicht hinterlegt zu sein. Dieser Sachverhalt ist mittig in Abbildung 3.2 zu sehen.

Abbildung 3.2 Das feindliche und das zivile Gebäude werden angegriffen.

3.4 Fazit

Abschließend möchte ich die wichtigsten Argumente für und gegen die Realitätsnähe des Spiels zusammenfassen. Da der Schwerpunkt meiner Arbeit insbesondere auf dem Einsatz von Luftfahrzeugen lag, kann ich mir in erster Linie nur ein Urteil über diese bilden. Besonders **positiv** zu bewerten sind:

- Die technischen Werte und Datenpakete, die den spielbaren Einheiten zugrunde liegen.
- Die Vielzahl an Möglichkeiten Höhe, Geschwindigkeit oder Kurs vorzugeben. Dies erhöht die taktische Wertigkeit von *C:MA/NO* und steigert ebenso das Gefühl alles beeinflussen zu können. Gewisse Vorhaben und Manöver lassen sich somit sehr genau simulieren.

Demgegenüber haben folgende Aspekte einen **negativen** Einfluss auf mein Fazit genommen:

- Die Tatsache, dass komplexe Operationen oder Einsätze unübersichtlich und für den Spieler nicht mehr zu kontrollieren sind. Die künstliche Intelligenz des Spiels vermag es nicht, taktische Anflüge oder taktisches Verhalten zu simulieren, sondern folgt relativ stur ihren vorgegebenen Routen und Befehlen. Dies erschwert dem Spieler die Kontrolle über seine Einheiten.
- Durch die umfangreiche Steuerung im Spiel geht meines Erachtens die Spieldynamik verloren. *C:MA/NO* hat den Anspruch, möglichst viele Handlungsalternativen abzubilden. Dadurch wirken die Szenarien programmiert. Man muss beispielsweise in den Emcon settings sehr viele Einstellungen vornehmen, bevor die Einheiten das gewünschte Verhalten gegenüber feindlichen Truppen zeigen.

Abschließend lässt sich sagen, dass sich das Spiel durch eine hohe Detailtiefe auszeichnet. Flugzeuge sind sehr realistisch dargestellt und operieren bis auf kleine Fehler nah an der Wirklichkeit. Andererseits ist jede Form von Infrastruktur (insbesondere Gebäude) im Spiel relativ schlecht abgebildet. Dies ist allerdings bei einer Software mit dem Namen *C:MA/NO* nicht weiter verwunderlich.

Nach meinen Erfahrungen sind bei Bodenangriffen erhebliche Schwächen des Spiels aufgetreten, daher besteht dort noch Nachbesserungsbedarf. Meines Erachtens zeichnet sich *C:MA/NO* durch einen hohen Selbstanspruch aus. Es möchte dem Spieler das Gefühl geben, alles beherrschen, kontrollieren und beeinflussen zu können. Dazu dienen die gut dargestellten Einheiten und die vielen Einstellungsmöglichkeiten. Insgesamt lassen sich Vor- und Nachteile aber nur schwer gegenüberstellen. Im Hinblick darauf wie realistisch *C:MA/NO* nun tatsächlich ist und ob es nun nur ein „Spiel“ ist komme ich zu folgendem Schluss:

Das Spiel zeigt gute Ansätze bezüglich der Realitätsnähe. Dennoch sind einige Dinge verbesserungswürdig (z.B. Infrastruktur und Geländedarstellungen). *C:MA/NO* bezeichnet sich selbst als Wargame. Allerdings wirken einige Manöver abstrakt und programmiert, wodurch die Spieldynamik erheblich eingeschränkt wird.

Folglich sehe ich in *C:MA/NO* kein typisches Computerspiel. Einige Ansätze wären wiederum charakteristisch für eine Simulation. Aber hier fehlt die wissenschaftliche Ausrichtung des Spiels. Es ist mit der Software keine statistische Auswertung der Szenarien möglich (z.B. einen Angriff 100 mal mit leicht veränderten Parametern durchführen lassen).

Bezüglich einer möglichen Nutzung innerhalb des Militärs lässt sich sagen, dass die sonst gute Detailtiefe für militärische Zwecke wiederum zu niedrig ist. *C:MA/NO* kann als *Allrounder-Spiel* verstanden werden, da es möglichst viele Operationen in vielen verschiedenen Umgebungen spielbar machen möchte. Dies führt allerdings zu einer gewissen Oberflächlichkeit, weshalb tiefgreifende und komplexe Simulationen mit militärischem Hintergrund nicht darstellbar sind. Vorstellbar wäre meiner Meinung nach aber schon, *C:MA/NO* auf strategischer Ebene für die Schulung militärischer Führungskräfte zu nutzen.

Schlussbemerkung

Alle hier beschriebenen Sachverhalte beruhen auf meinen Erfahrungen mit dem Spiel *Command: Modern Air/Naval Operations*. Sämtliche Bilder sind diesem Spiel entnommen.

Kapitel 4
Luftmanöver nahe Norwegen

Clemens Vosseberg

Zusammenfassung Im Rahmen der Untersuchung des Computerspiels *Command: Modern Air/Naval Operations* (*C:MA/NO*) wurde ein Szenarien erstellt, um weitere Erkenntnisse über die Modellbildungsfähigkeit des Spiels zu gewinnen. Dieses Szenario ist an Übungen von russischen Bomberverbänden in der Nähe des NATO Luftraums aus dem Jahre 2014 angelehnt.

4.1 Szenario Aufbau

Im Folgenden wird auf den generelle Aufbau des Szenarios, sowohl auf Seite der NATO als auch auf russischer Seite eingegangen.

4.1.1 NATO-Seite

Beim Erstellen des Szenarios wurden zunächst die norwegischen Luftradarstationen anhand der realen GPS Standorte in *C:MA/NO* eingefügt.

Folgende norwegische Radarstützpunkte[1] sind in dem Szenario vorhanden:

1. Mågerø (70°59'3"N, 25°53'49"E) Control Reporting Center
2. Mt Gråkallen (63°25'12"N, 10°15'3"E) remote controlled AD radar
3. Mt Njunis (68°43'54"N, 19°32'4"E) remote controlled AD radar
4. Kirkenes airport(69°42'57"N, 29°52'10"E) remote controlled AD radar
5. Klettkovfjellet (67°10'23"N, 15°1'41"E) remote controlled AD radar

Clemens Vosseberg
Helmut-Schmidt-Universität/Universität der Bundeswehr Hamburg, Holstenhofweg 85, 22043 Hamburg, e-mail: clemens.vosseberg@hsu-hh.de

[1] Quelle: https://en.wikipedia.org/wiki/Integrated_NATO_Air_Defense_System

6. Selbu (63°18’41”N, 10°56’18”E) remote controlled AD radar
7. Sørreisa (69°8’11”N, 18°8’14”E) Control Reporting Center
8. Vågsøy (61°59’53”N, 5°3’1”E) remote controlled AD radar
9. Vardø (70°22’0”N, 31°7’35”E) remote controlled AD radar

Im Szenario werden diese Stützpunkte von *C:MA/NO* durch das „HADR SINDRE I“ Radar abgebildet.

Des weiteren wurden die norwegischen Luftwaffenstützpunkte[2], auf denen derzeit F-16 Kampfflugzeuge stationiert sind, dargestellt:

1. Bodø hovedflystasjon (67°16’9N, 14°21’55”E)
2. Ørland hovedflystasjon (63°41’57N, 9°36’14”E)

4.1.2 Russische Seite

Auf russischer Seite wurde ein Bomberverband, bestehend aus TU-95 Langstreckenbombern und Il-78 Betankungsflugzeugen, eingefügt (siehe Abb. 4.1, Szenario ①). Dieser fliegt anhand einer Support-Mission die norwegische Küste ab[3]. Zudem wurde eine No Navigation Zone eingerichtet (rote Fläche in Abb. 4.2), welche verhindern soll, dass beim Rückflug der NATO-Luftraum durchquert wird.

4.2 Szenario Ablauf und Beobachtungen

Nach dem Aufbau der Mission wurde das Szenario wiederholt durchlaufen, um die Fähigkeiten des Spiels eingehend zu untersuchen.

4.2.1 Ablauf

Gemäß der vorgegebenen Route fliegt der russische Verband entlang der Küste, bis er durch die Radarsysteme aufgeklärt wird. An diesem Punkt wird der Spieler aktiv und startet eine Air Intercept Mission vom norwegischen Luftwaffenstützpunkt Bodø hovedflystasjon aus. Die F-16 Kampfflugzeuge starten im Anschluss, fliegen selbständig auf den Bomberverband zu, passen sich deren Fluggeschwindigkeit an und folgen ihm, bis ihre maximale Flugzeit erreicht ist. Im Anschluss treten sie selbständig den Rückweg zur Basis an.

---

[2] Quelle: https://de.wikipedia.org/wiki/Norwegische_Luftstreitkräfte

[3] Quelle: Nzz.ch (http://www.nzz.ch/international/europa/russische-muskelspiele-1.18415257)

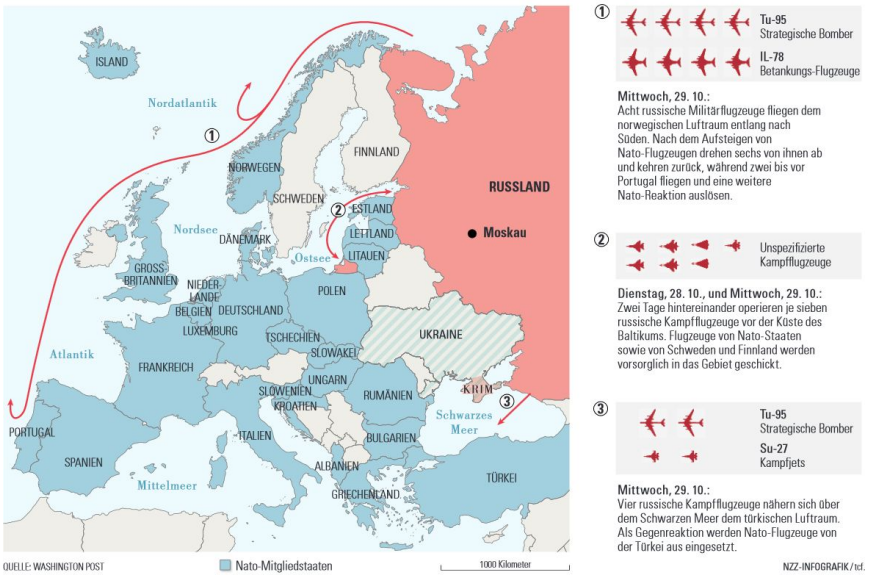

Abbildung 4.1 Russische Luftmanöver über Europa, Szenario ①, ② und ③. Das Szenario wurde in Anlehnung an ① erstellt.

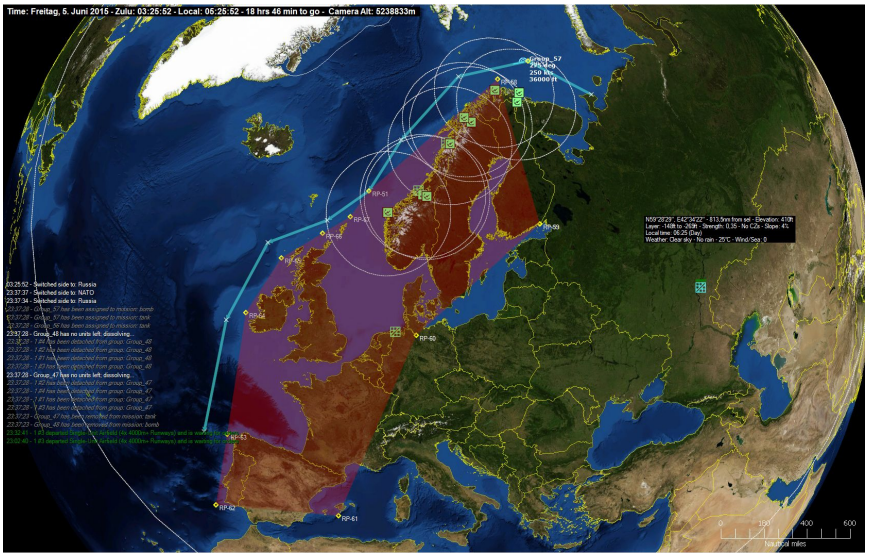

Abbildung 4.2 Szenario Übersicht mit God's Eye view (Anzeigen aller Einheiten auf beiden Seiten).

4.2.2 Beobachtungen

Es wurde festgestellt, dass die Flughöhe einen entscheidenden Einfluss auf die Radaraufklärung hat. Je größer die Flughöhe ist, in der operiert wird, desto eher werden die Flugzeuge aufgeklärt. In niedrigen Höhen ist es daher möglich die Radaraufklärung zu umgehen und unentdeckt an der Küste zu operieren. Die No Navigation Zones werden zwar bei der Missionserstellung ausgeschlossen, jedoch beim Rückflug teilweise an den Rändern durchflogen. Dennoch ist eindeutig erkennbar, dass die Lufteinheiten eine andere Route wählen und die Eckpunkte der Zone umfliegen, anstatt wie zuvor den kürzesten Weg zur Heimatbasis zu wählen. Beim wiederholten Testen des Szenarios ist zudem aufgefallen, dass es manchmal noch zu Fehlern im Spielablauf (Bugs) kommt. So verschwanden Flugzeuge ohne Absturz- oder Abschussmeldung und waren auch auf dem zugewiesenen Flugplatz nicht aufzufinden. Ob es sich hier lediglich um eine fehlenden Ausgabe von Ereignissen im Message Log oder um ein tatsächliches Verschwinden der Einheiten handelte, konnte jedoch nicht sicher ermittelt werden. Ebenfalls kam es während des Modellierens und Spielens mehrfach zu Fehlermeldungen und Spielabstürzen.

4.2.3 Flugverhalten

Bei der Erstellung und der anschließenden Auswertung des Szenarios ließen sich auch Erkenntnisse über das modellierte Flugverhalten der Lufteinheiten gewinnen.

Flugzeuge haben in *C:MA/NO* einen höhenabhängigen Treibstoffverbrauch. In niedrigen Höhen wird mehr Treibstoff verbraucht als in hohen. Die Modelle der Luftfahrzeuge kalkulieren ihren Rückflugweg mit den entsprechend günstigen Flughohen und wechseln auf dem Rückweg automatisch auf diese. Dies geschieht auch dann wenn der Spieler zuvor manuell eine andere Flughöhe vorgegeben hat. Abbildung 4.3 zeigt die Kraftstoffverbräuche in Abhängigkeit von der Flughöhe. Diese wurden exemplarisch für einen russischen Bomber Typ TU-95 Bear H ermittelt.

Es wird deutlich, dass der Tiefflug einen fast dreimal so hohen Treibstoffverbrauch nach sich zieht, wie der standardmäßige Höhenflug. Dies schränkt die Reichweite massiv ein.

Neben der Flughöhe hat auch die Fluggeschwindigkeit einen entscheidenden Einfluss auf den Treibstoffverbrauch (Abb. 4.4).

4.3 Weitergehende Untersuchungen

Im weiteren Verlauf kam der Wunsch auf anhand der Szenarien weitere Fragestellungen zu bearbeiten.

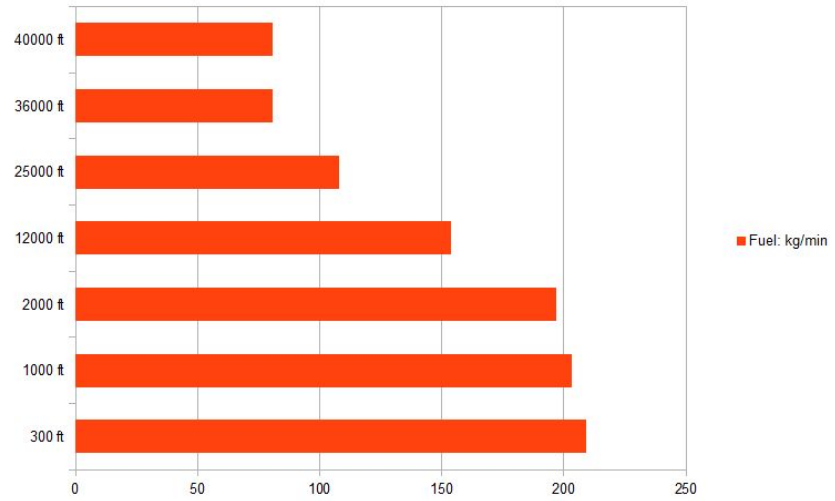

Abbildung 4.3 Kraftstoffverbrauch in Abhängigkeit von der Höhe, Angaben in kg Kerosin pro Minute.

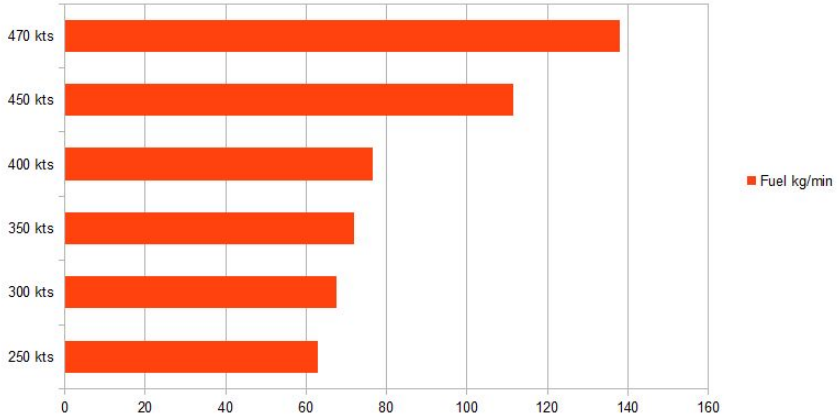

Abbildung 4.4 Kraftstoffverbrauch in Abhängigkeit von der Geschwindigkeit, Angaben in kg Kerosin pro Minute (Höhe 40.000 ft).

4.3.1 Out of fuel

Eine der Fragen war, was passieren würde, wenn der Treibstoffvorrat eines Flugzeuges zur Neige ginge. Sollte der Treibstoff eines Flugzeuges aufgebraucht seien, so hat dies einen Absturz zur Folge. Der Spieler wird hierüber akustisch, durch ein Absturzgeräusch und eine Konsolenmeldung im Message Log informiert. Hier zeigt sich jedoch wieder eine große Schwäche von *C:MA/NO* als Analysewerkzeug, da die Koordinaten bei der Meldung nicht mit ausgegeben werden. Eine Recherche des Absturzortes muss daher aufwändig manuell über die Aufnahmefunktion ermittelt werden oder noch während des Spiels recherchiert werden.

```
21:13:45 - 1 3 has been destroyed!
21:13:45 - 1 3 (Il-78 Midas) has run out of fuel and crashed!
```

Abbildung 4.5 Ausgabe im Message Log beim Absturz einer Il-78.

4.3.2 Einsatz von EF 2000 aus England

Nach Aufklärung der Bomberverbände durch die Radarstationen (gelber Korridor) brauchen die F-16 nach dem Start im Schnitt ca. 55 Minuten Flugzeit, um den russischen Verband zu erreichen (grüner Korridor). Anschließend begleiten sie den gegnerischen Verband, wobei die Geschwindigkeit hierbei automatisch angepasst wird. Nach ca. 2:30 Stunden Gesamtflugzeit wird der Rückflug eingeleitet (roter Korridor). In diesem Bereich müssen andere Kampfflugzeuge den Verband übernehmen. Da die Entfernung für den Einsatz englischer Abfangjäger zu groß ist, mussten erneut Norwegische F-16 den Begleitflug sicherstellen.

Die zweite Rotte folgt dem Verband nun, bis ihre Flugzeit im weißen Korridor endet. Ab hier können englische Jagdflieger (Typ Tornado) von der RAF Basis Lossiemouth[4] aus den Verband übernehmen und weiter begleiten (siehe Abb. 4.6).

4.3.3 Radar Gaps

Zur Thematik der Radar Gaps sei an dieser Stelle auf das Kapitel 5 verwiesen, welches sich intensiv mit der Abbildung der Radarsysteme in *C:MA/NO* beschäftigen. Die dortige Grunderkenntnis, dass mit zunehmender Flughöhe der Flugzeuge sich diese immer früher auffassen lassen, kann an dieser Stelle bestätigt werden. Bei

[4] Quelle: https://en.wikipedia.org/wiki/RAF_Lossiemouth

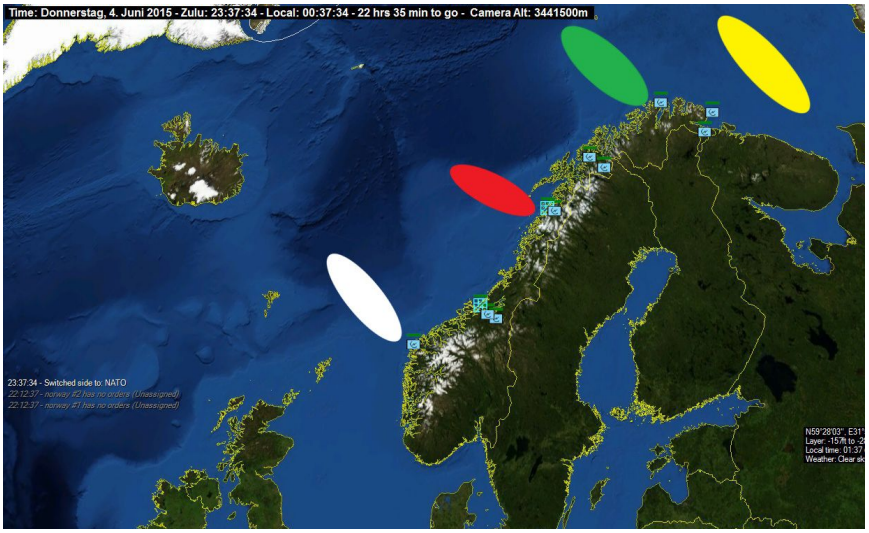

Abbildung 4.6 Übergabegebiete

einer Anflughöhe von 300 Fuß (ft) werden die Bomber erst in einer Entfernung von ca. 6 nautischen Meilen (NM) aufgefasst. Dies erklärt auch die zuvor genannte Möglichkeit, die Radarerfassung im Tiefflug zu umgehen.

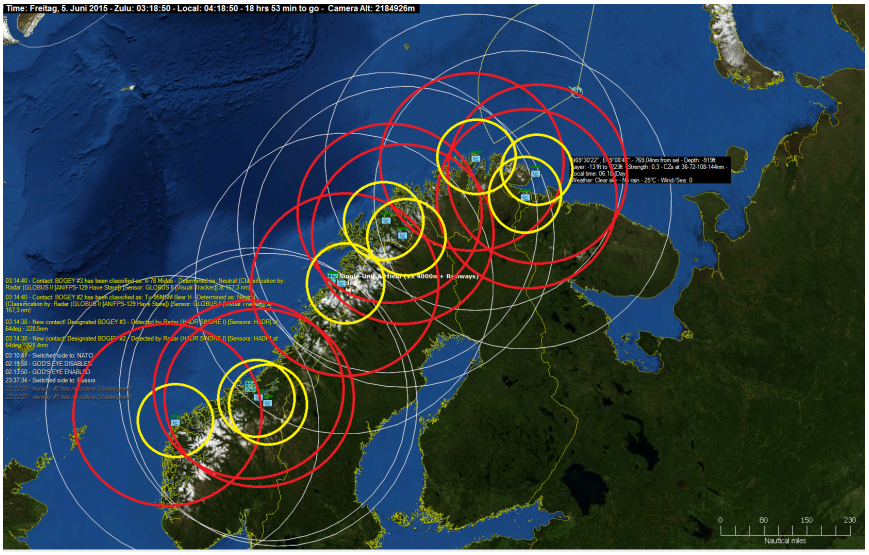

Abbildung 4.7 Radarerfassung bei 25.000 ft und 10.000 ft.

Die norwegische Luftverteidigung beginnt bei Anflughöhen ab ca. 10.000 ft lückenhaft zu werden. Exemplarisch ist die Radarabdeckung bei 25.000 ft (roter Kreis) und bei 10.000 ft (gelber Kreis) eingezeichnet (siehe Abb. 4.7). Die weißen Kreise markieren die höchstmögliche Reichweite (270 NM).

Kapitel 5
Untersuchung des *C:MA/NO*-Radar-Modells

Simon Funke

Zusammenfassung Ziel der folgenden Untersuchungen ist es, eine möglichst exakte Vorstellung davon zu erlangen, wie *C:MA/NO* die Sensoren einzelner Waffensysteme, im speziellen die Radarsysteme, abbildet und zu prüfen, ob dies auch der Realität sehr nahe kommt. Zu diesem Zweck habe ich zum Einen selbst erstellte Szenarien simuliert und anschließend die Beobachtungen und Ergebnisse genau dieser Fragestellung unterzogen. Zum Anderen habe ich ein vom Entwickler vorgegebenes Szenario gespielt. Des Weiteren habe ich meine persönlichen Erfahrungen aus diesen Untersuchungen, welche ich mit dem Spiel gemacht habe, im folgenden Bericht dargestellt.

5.1 Erster Kontakt mit dem Spiel

Mein erster Kontakt mit dem Spiel war im Rahmen des ISA-Studienkurses im Mai 2015. Zunächst wurden wir von Prof. Dr. Fügenschuh in die Tutorials des Spiels eingewiesen, um eine erste Vorstellung von dem Spiel zu bekommen und die ersten Schritte im Umgang mit der Software zu erlernen.

Im ersten Moment stellte ich fest, dass die Spieloberfläche nicht den Spielen ähnelt, welche aktuell auf dem Markt erscheinen. Sie ähnelte vielmehr einer taktischen Simulationssoftware, in der sich lediglich Symbole bewegen, jedoch keine umfangreichen grafischen Animationen mehr stattfinden, wie man es heutzutage von Spielen gewohnt ist.

Bei den Tutorials, die uns der Professor vorstellte, handelte es sich um zwei maritime Operationen und eine Luftoperation. Im ersten Tutorial war es die Aufgabe, mit der Hilfe eines U-Bootes ein feindliches U-Boot ausfindig zu machen und zu bekämpfen.

Simon Funke
Helmut-Schmidt-Universität/Universität der Bundeswehr Hamburg, Holstenhofweg 85, 22043 Hamburg, e-mail: Simon.Funke@hsu-hh.de

Das Spiel wirkte zu Beginn sehr unübersichtlich und überaus umfangreich. Zudem ist die Spieloberfläche komplett in englischer Sprache. Dennoch war es zu Anfang sehr schnell möglich, einfache Abläufe, wie zum Beispiel das Koordinieren von Einheiten, oder die Zuweisung und Erstellung verschiedener Missionen zu erlernen und die Benutzeroberfläche besser zu verstehen.

Den ersten Missionstyp, den man hier erlernte, war die sogenannte *Patrol-Mission*. Wie der Name schon verrät, handelt es sich hierbei um eine Mission, welche zur Aufgabe hat, ein bestimmtes Gebiet zu patrouillieren. Dazu muss man Punkte auf der Karte markieren, sogenannte *Ref.points*, welche man zu einem Auftragsgebiet zusammenfassen kann. In dem Missionsmenü kann man daraufhin seine Einheiten den jeweiligen Missionen zuweisen und ihnen so die Aufgabe erteilen, das markierte Gebiet zu durchstreifen.

Die Tutorials erwiesen sich als durchaus nützlicher Leitfaden, der mir dabei geholfen hat, diese ersten Eindrücke und Abläufe schnell zu verstehen.

Um danach die ersten Missionen zu spielen, bedurfte es keiner neuen Tutorials mehr. Der Umgang mit dem Interface war schneller erlernt als zunächst gedacht und die komplexe Informationsflut schnell entschlüsselt. So konnte ich nach circa 10 Spielstunden feststellen, dass die Sprache und die Bedienung nur noch bedingte Hindernisse darstellten. Ebenso erlernte man mit jedem neuen Szenario neue Befehle und die Nutzung anderer Missionstypen. Das englische Vokabular eignete ich mir automatisch mit jedem neuen gespielten Szenario an.

Im Großen und Ganzen war der Erstkontakt mit der Software sehr fordernd, was sich jedoch sehr schnell in Verständnis und Verstehen weiterentwickelte.

5.2 Szenario 1: *Falklands War 1982, Sea of Fire*

Das erste Szenario war *The Falklands War*. Inhaltlich spielt dieses Szenario, angelehnt an den tatsächlich stattgefundenen Falklandkrieg, vor der Küste Argentiniens zum Ende des 20. Jahrhunderts. Das Durchlaufen des Falkland-Szenario diente einerseits dem Erlernen und Ausbauen weiterer spielerischer Taktiken und Tricks, und andererseits stellte es schon den ersten analytischen Teil der Grundfrage dar, welchen ich mir zu Beginn gestellt habe.

5.2.1 Hintergrund und Mission

Im 16. Jahrhundert wurden die Falklandinseln zum ersten Mal entdeckt. Seitdem sind durch zahlreiche Machtwechsel und verschiedene Besetzungen der Inseln ebenso unterschiedliche Besitzansprüche verschiedener Nationen, hier besonders Argentinien und Großbritannien, entstanden. Am 02.04.1982 wollte Argentinien seine Besitzansprüche gegenüber anderen Nationen durchsetzen und landete dazu Truppen auf der Inselgruppe. Großbritannien intervenierte und wollte diesen An-

spruch nicht anerkennen. Die Jahrzehnte andauernden politischen Interessen beider Nationen an den Inseln hatten sich so zu einem bewaffneten Konflikt entwickelt.

In dem Spiel war es nun der Auftrag, als argentinischer Verteidiger die Inseln vor zwei naheliegenden Kriegsschiffen der Royal Navy zu schützen und mindestens eines der Kriegsschiffe zu zerstören. Zur Verfügung hatte ich zwei Bomber, Super Entard, der argentinischen Navy und zwei Jägergruppen mit jeweils drei Flugzeugen des Typs A-4 Skyhawk (siehe Abbildung 5.1).

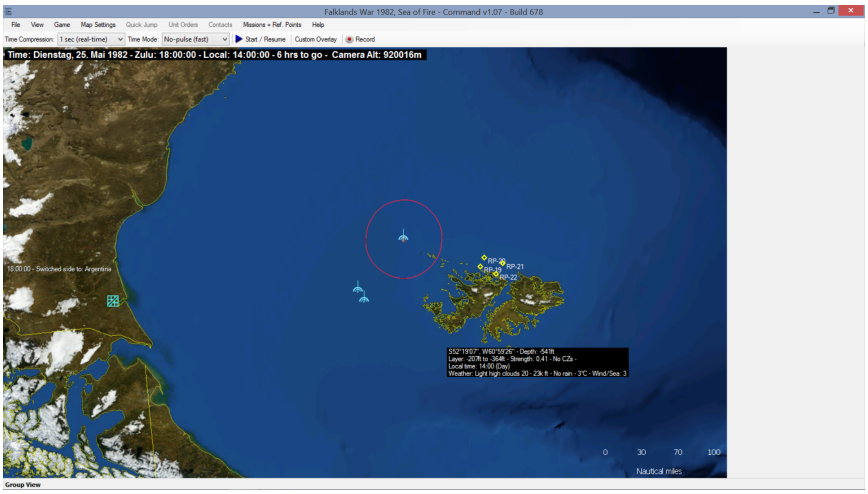

Abbildung 5.1 Szenario *Falklands War*.

5.2.2 Herangehensweise und Durchführung

In einem Briefing erfährt man vorab, dass die sehr knapp bemessenen Treibstofffüllstände der Flugzeuge berücksichtigt werden sollen. Dazu konnte man die Flugzeuge, indem man sie kurz vor Erreichen ihres Treibstofflimits einer *Strike-Mission* zuwies, noch dazu bringen, alle verfügbaren Waffensysteme abzufeuern um so das Gewicht des Flugzeugs und den daraus resultierenden Treibstoffverbrauch für den Rückflug zu verringern.

Der Erstversuch

Beim ersten Herangehen an dieses Szenario habe ich, ohne mich weiter über den Konflikt oder die Waffensysteme zu informieren, meine bis dahin erlernten Fähig-

keiten abgerufen und zunächst eine *Patrol-Mission* erstellt. Dazu habe ich die Box in der Karte, in welcher sich die Schiffe aufhalten sollten, markiert. Den Flugzeugen konnte ich durch eine *Patrol-Mission* die oben genannte Aufgabe zuweisen. Ziel dabei war es vorerst zu beobachten, wie der Gegner und die eigenen Kräfte agieren und reagieren um eine erste Vorstellung zu bekommen, wie dieses Szenario genau abläuft und gegebenenfalls Anpassungen vorzunehmen.

Das Erste, was zu beobachten war, dass die Jäger abdrehten, bevor sie die Kriegsschiffe aufklären konnten. Sie nahmen keine aktive Rolle in diesem ersten Durchlauf ein. Die Bomber haben die Kriegsschiffe sehr schnell aufgeklärt und ohne zu zögern direkt ihre Flugkörper auf die Ziele abgefeuert. Dabei war zu beobachten, dass die Bomber sehr schnell von den Schiffen bekämpft wurden, genauso wie die sich annähernden Flugkörper. Überraschend war, dass ein Flugkörper zur Wirkung gekommen ist und ein Schiff zerstört wurde.

Das Fazit lautet hier, dass die Mission erfüllt war, da ein Schiff erfolgreich bekämpft wurde. Dennoch blieb es ein persönliches Ziel, alle Flugzeuge aktiv in das Geschehen mit einzubeziehen.

Der Zweitversuch

Im Zweitversuch wurde die Herangehensweise von Grund auf verändert. So habe ich mich an erster Stelle mit dem Hintergrund des Konflikts beschäftigt und einiges über die geografische Struktur der Inseln erfahren, was mir vorher gänzlich unbekannt war. So konnte ich herausfinden, dass die Inseln einen hohen Gebirgszug aufweisen, welchen ich im Folgeversuch nutzen wollte, um nicht später vom feindlichen Radar aufgefasst zu werden. Dazu habe ich die bisherigen Flughöhen der Flugzeuge betrachtet und festgestellt, dass diese auf Maximalhöhe (40.000 ft) fliegen. Auch hier wollte ich variieren und testen, ob es möglich ist, das feindliche Radar zu unterfliegen. Außerdem habe ich die Waffensysteme genauer betrachtet, um allgemein einen Eindruck der möglichen Fähigkeiten zu erhalten. Hier bekam ich eine Impression der Primärbewaffnung meiner Flugzeuge. Zur Durchführung des zweiten Versuchs habe ich mir die Hilfestellung, welche im Missionsbriefing geschildert wurde, zur Hilfe genommen und meinen Flugzeugen genaue Flugrouten zugewiesen. Die *Patrol-Mission* blieb bestehen, nur der Anflug hatte sich im Aspekt der Flughöhe und der Anflugrichtung verändert. Die Bomber sind, ausgenommen der Flughöhe, unverändert auf die Schiffe zugeflogen und haben ihre Flugkörper abgefeuert. Es folgte, dass beide Flugzeuge genauso früh erkannt wurden, wie beim ersten Versuch, und gleichermaßen wie ihre Flugkörper abgeschossen wurden.

Die Jägergruppen haben, anders als beim Erstversuch, die Schiffe erreicht. Sobald sie einen bestimmten Abstand zu den Schiffen hatten, haben sie um die Kriegsschiffe Kreise gezogen und wurden nacheinander bekämpft. Währenddessen haben die Flugzeuge keine einzige offensive Maßnahme ergriffen.

Das Fazit lautet hier, dass die Mission gescheitert war. Einen Teilerfolg konnte ich für mich, jedoch nicht für die Mission, erzielen, indem ich es geschafft habe,

anders als beim Erstversuch, alle Flugzeuge in das Kampfgeschehen einzubeziehen. Warum die Flugzeuge genauso früh aufgefasst wurden, bleibt unklar. Auch das Überfliegen des Gebirges hat nicht zu dem erhofften Erfolg geführt.

Die Folgeversuche

Durch die daraus gewonnenen Erfahrungen habe ich für die Folgeversuche beschlossen, Anflughöhen, Anflugrichtungen und Geschwindigkeiten zu variieren. Zusätzlich habe ich eine *Strike-Mission* genutzt, um meine Flugzeuge dazu zu bringen, alle möglichen Waffen zu nutzen, die ihnen zur Verfügung standen, bevor sie zur Basis zurückkehrten. Dies habe ich aufgrund der Tipps in dem Briefing genutzt. Bei der *Strike-Mission* muss man, anders als bei der *Patrol-Mission*, den eigenen Einheiten ein Ziel vorgeben, welches man markiert hat. Diese werden anschließend das markierte Ziel angreifen. Die Voraussetzung für eine solche Mission ist, dass die feindliche Einheit aufgeklärt wurde, andernfalls kann man sie nicht markieren und keine *Strike-Mission* erstellen. Ebenso versuchte ich die Flugzeuge so zu steuern, dass sie nahezu gleichzeitig bei den Kriegsschiffen eintrafen, um eine Überforderung dieser zu bewirken. Zuerst war zu beobachten, dass die Flugzeuge alle eine unterschiedliche Reichweite bei unterschiedlichen Flughöhen und Geschwindigkeiten aufwiesen.

So hat sich beispielsweise der Treibstoffverbrauch bei hohen Geschwindigkeiten erhöht und bei langsameren verringert. Ebenso hat er sich in hohen Flughöhen verringert und in niedrigen erhöht.

Das Ergebnis der Folgeversuche war nahezu immer gleich. Es wurde maximal ein Schiff zerstört und die Jäger haben niemals auf Schiffe gefeuert. Somit waren ausnahmslos die zwei Flugkörper das Einzige, womit ich die Schiffe aktiv bekämpfen konnte.

5.2.3 Fazit

Die Mission ist nach der Vorgabe erfüllbar. Jedoch erweisen sich *Air-Operations* als sehr umfangreich und schwer in der Koordination. Im Bezug auf die Geographie konnte man in diesem Szenario lediglich feststellen, dass Gebirge vorhanden sind und als solche auch mit unterschiedlichen Höhen abgebildet werden. Allerdings erwiesen sie sich hier nicht als hilfreich, was diese Frage, ob Gebirge einen Effekt auf Sensoren haben, für zukünftige Szenarien noch unbeantwortet lässt.

Variationen in den Flughöhen und in der Geschwindigkeit zeigten Auswirkungen auf die Flugdauer und Reichweite der Flugzeuge. Diese ließen jedoch keine erkennbaren Unterschiede in der Auffassung durch das feindliche Radar erkennen, was die Grundfrage somit leider nicht beantwortete. Es ließ nur die erste Vermutung offen, dass das Radar, meinen Auswertungen nach, nicht der Realität gleicht. Es gilt nun

in den folgenden Szenarien diese Fragestellung zu beantworten. Die Beobachtung des Treibstoffverbrauchs der einzelnen Flugzeuge mit verschiedenen Geschwindigkeiten und Höhen lässt die Vermutung zu, dass die Spielentwickler hier versuchen, realistische Gegebenheiten mit in die Software einzubeziehen.

5.3 Szenario 2: *Radarauffassung der Fregatte Klasse 124*

Dieses Szenario habe ich als Folgeszenario entworfen, um abgeleitete Fragestellungen aus dem ersten Szenario tiefergehend zu untersuchen. Das zweite Szenario, welches ich, im Gegensatz zum Ersten, selbst erstellt habe, soll nun aufzeigen, ob unterschiedliche Flughöhen und unterschiedlich große Flugzeuge auch dementsprechend früher oder später vom Radar aufgefasst werden. Folglich gehe ich wie gehabt auf die Grundfrage ein, ob diese Software die Sensoren einzelner Waffensysteme, im speziellen die Radarsysteme, abbildet und ob dies auch der Realität nahe kommt.

5.3.1 Beschreibung

Als Waffenplattform wählte ich die Fregatte „Sachsen“ der Klasse 124, weil diese mit der „SMART-L“ über ein sehr leistungsstarkes Radarsystem verfügt[1].

Das Schiff habe ich auf der Karte mitten im Atlantischen Ozean platziert. Der Grund dafür war, dass unterschiedliche Gelände, wie zum Beispiel Gebirge, keine Auswirkungen auf das Radar haben sollten. Ebenso wurde das Wetter auf klare Sicht, Sonnenschein und keine Bewölkung eingestellt. Im weiteren Ablauf sollten nun zwei verschiedene Flugzeuge, verschiedener Größe, die Fregatte anfliegen. Als kleines Flugzeug wählte ich den Eurofighter und im Gegensatz dazu als großes Flugzeug den Airbus A400M (siehe Abbildung 5.2).

Der Fregatte wurde zu Beginn einer *Patrol-Mission* zugewiesen, mit der Aufgabe, in einem sehr kleinen Gebiet nach Flugzeugen zu suchen. Die Flugzeuge bekamen als Mission nur einen *Transit (Ferry)* zugewiesen. Bei diesem Missionstyp fliegen die Einheiten eine feste und vorgegebene Route ab. Demzufolge stellte sich hier die Frage, wann die Fregatte die Flugzeuge auffasst. Besonders dabei spielt es eine wichtige Rolle, wie der Radar in der Software dargestellt wird. Zu sehen ist nur ein weißer Kreis mit großem Abstand um die Fregatte herum. Dieser stellt die maximale Reichweite des Radars dar (siehe Abbildung 5.3).

Angenommen, dass dieses Spiel den Radar als Zylinder darstellt, müsste unabhängig von der Flughöhe oder Flugzeuggröße, jedes Flugzeug am Rande des weißen Kreises aufgefasst werden. Andernfalls würden sich Änderungen der Parameter bemerkbar machen. So müsste ein kleines Flugzeug später aufgefasst werden und ein niedriger fliegendes ebenfalls.

[1] Siehe www.kriegsschiffe.eu, abgerufen am 17.06.2015.

Abbildung 5.2 Optischer Vergleich Eurofighter - Airbus A400M (http://earth66.com/img/Airbus-A400M-Dassault-Rafale-and-Eurofighter-Typhoon-during-the-2013-Bastille-Day-parade.jpg).

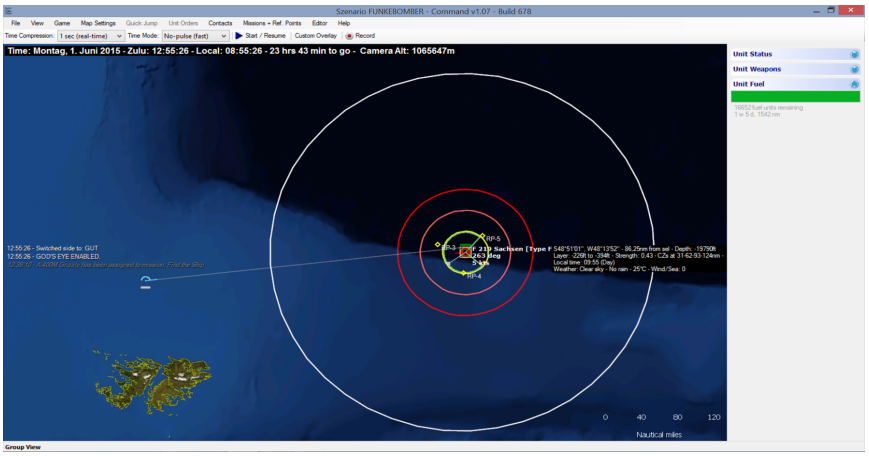

Abbildung 5.3 Szenario 2.

5.3.2 Versuchsdurchlauf Airbus A400M

Im ersten Durchlauf habe ich den Airbus A400M anfliegen lassen, um die Daten zu sammeln, wann dieser frühestens aufgeklärt wird. Dazu bin ich bei 40.000 ft gestartet und habe für jeden weiteren Durchlauf die Höhe im Abstand von 2000 ft reduziert. Dabei habe ich beobachtet, dass bei einer Flughöhe von 40.000 ft bis 28.000 ft keine Änderung bei der Auffassung bemerkbar wurde. Der Airbus wurde

nahezu jedes Mal bei einer Distanz von ca. 214 nautischen Meilen, mit geringfügigen Abweichungen, erfasst (siehe Abbildung 5.4).

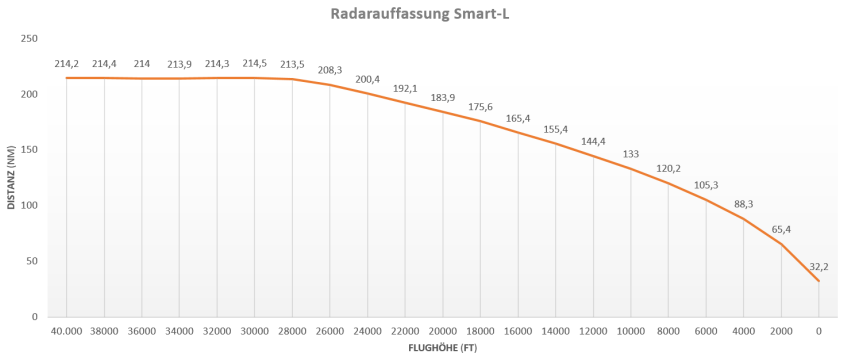

Abbildung 5.4 Erfassungsprofil Airbus A400M.

Dies entspricht nicht der maximalen Reichweite des Radars, was die erste Frage, ob der Radar als Zylinder dargestellt wird, beantwortet. Offensichtlich ist die maximale Reichweite nicht die Begrenzung für die Auffassung des Radars.

In den folgenden Durchläufen zeigte sich, dass der Airbus mit sinkender Flughöhe auch später entdeckt wurde. Im Vergleich wird der Airbus bei einer Flughöhe von 300 ft, was der Mindesthöhe entspricht, erst bei einer Distanz von circa 32 nautischen Meilen zur Fregatte erfasst. Diese Beobachtung lässt schon frühzeitig darauf schließen, dass die Flughöhe Auswirkungen auf die Erfassung durch das Radar hat.

5.3.3 Versuchsdurchlauf Eurofighter

Im zweiten Durchlauf dieses Szenarios ließ ich den Eurofighter, genau wie den Airbus, aus den verschiedenen Höhen anfliegen, auch beginnend bei 40.000 ft. Der Eurofighter wurde frühestens bei einer Distanz von circa 145 nautischen Meilen aufgeklärt. Das veränderte sich auch nicht bis zu einer Höhe von 12000 ft. Erst bei Flughöhen darunter wurde er später aufgefasst. Bei seiner Mindestflughöhe von 80 ft wurde er erst bei einer Distanz von 20 nautischen Meilen aufgeklärt (siehe Abbildung 5.5).

Dieses Resultat bestätigt meine Annahme aus dem ersten Durchlauf mit dem Airbus A400M. Die Radarauffassung entspricht demzufolge nicht der Abbildung eines Zylinders, und ebenso variiert mit der Flughöhe auch die Auffassungsfähigkeit des Radars.

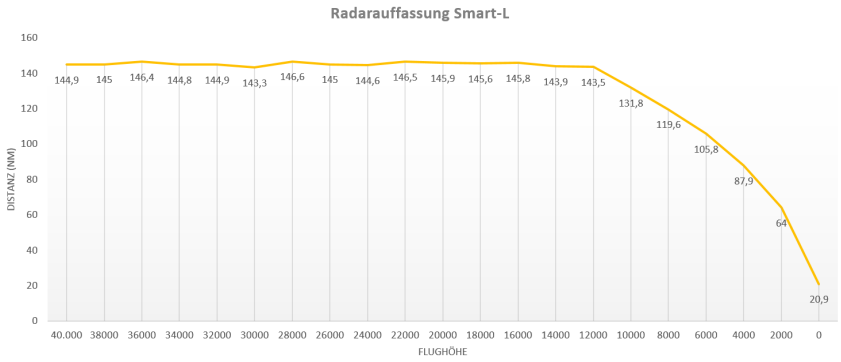

Abbildung 5.5 Erfassungsprofil Eurofighter.

5.3.4 Vergleich

Im Vergleich der beiden erstellten Profile wird deutlich, dass der Airbus viel früher aufgefasst wird als der Eurofighter. Jedoch folgen sie grafisch ab einer Flughöhe von 12.000 ft den gleichen Distanzen.

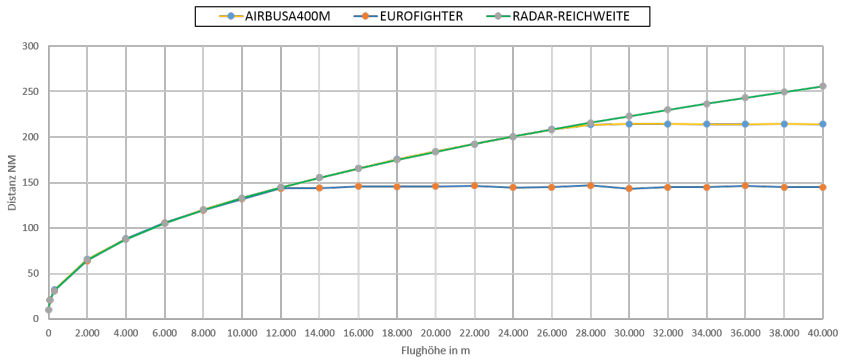

Abbildung 5.6 Vergleich der erstellten Profile und der *Line of Sight*.

Sie werden demnach, unabhängig von der Größe des Flugzeugs, unterhalb einer Flughöhe von 12.000 ft zeitgleich aufgefasst (siehe Abbildung 5.6). Die Größe spielt demzufolge nur in für darüber liegende Flughöhen eine Rolle.

Hier grün dargestellt ist die *Line of Sight*[2], die direkte Sichtlinie zwischen dem Radar und den Flugobjekten, unter Einberechnung der Erdkrümmung.

[2] Siehe www.gitta.info/TerrainAnaly/de/html/OpforEngAppl_learningObject1.html, abgerufen am 15.07.2015.

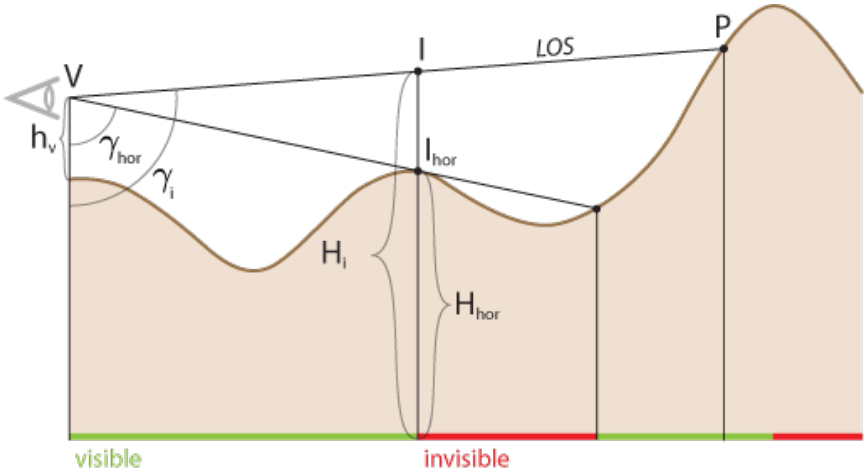

Abbildung 5.7 Line of Sight (www.gitta.info/TerrainAnaly/de/html/OpforEngAppl_learningObject1.html).

Legt man die berechnete *Line of Sight* (siehe Abbildung 5.7) über die Ergebnisse der erstellten Erfassungsprofile, wird deutlich, dass dieses Spiel den physikalischen Gegebenheiten in diesem Punkt gerecht wird. Die Flugzeuge können frühestens in der *Line of Sight* erkannt werden. Das stellt der Bereich unterhalb des grünen Graphen dar. Der Abschnitt der oberhalb dieser Linie liegt, befindet sich auch außerhalb der Sichtlinie. Würde man hier Ergebnisse in den Profilen erkennen, müsste man feststellen, dass dieses Spiel nicht die Erdkrümmung und die daraus resultierenden Einschränkungen für die Radarauffassung abbildet.

5.3.5 Fazit

Die Software berücksichtigt bei der Darstellung von Radarsystemen bezüglich der Flugzeuggrößen, dass größere Flugzeuge früher erkannt werden als kleinere. Ebenso beachten die Entwickler, dass die Radarstrahlung von der *Line of Sight* in ihrer Reichweite begrenzt wird.

Die letzte Frage, welche noch nicht beantwortet wurde, muss sich demnach damit beschäftigen, wie der Unterschied in der Auffassung für verschieden große Flugzeuge berechnet und abgebildet wird. Folglich, ob die Energie, die der Eurofighter vor einer Distanz von 145 nautischen Meilen zurückstrahlt, zu gering ist, um erfasst zu werden, ob es einem anderen Muster folgt oder ob es sich um willkürlich gewählte Werte handelt.

5.4 Szenario 3: AWACS

In diesem Szenario habe ich mich dazu entschlossen nicht die Fragestellung der Energieabstrahlung einzelner Flugzeuge zu untersuchen, sondern stattdessen zu prüfen, ob die Software auch in der Lage ist ein mobiles Radar, eine Boeing E-3, welche zur Luftraumüberwachung genutzt wird, abzubilden (siehe Abbildung 5.9).

5.4.1 Beschreibung

Eine Boeing E-3 zählt zu den sogenannten *Airborne Early Warning and Control System*, kurz *AWACS*. Diese Flugzeuge haben den Zweck, den Luftraum aus einer Höhe von ca 10.000 Meter zu überwachen. Dabei strahlen sie, anders als ein bodengebundenes Radar, den Radarkegel von oben nach unten ab, und erhalten so eine viel breitere Überwachungsfläche zum Boden hin.[3]

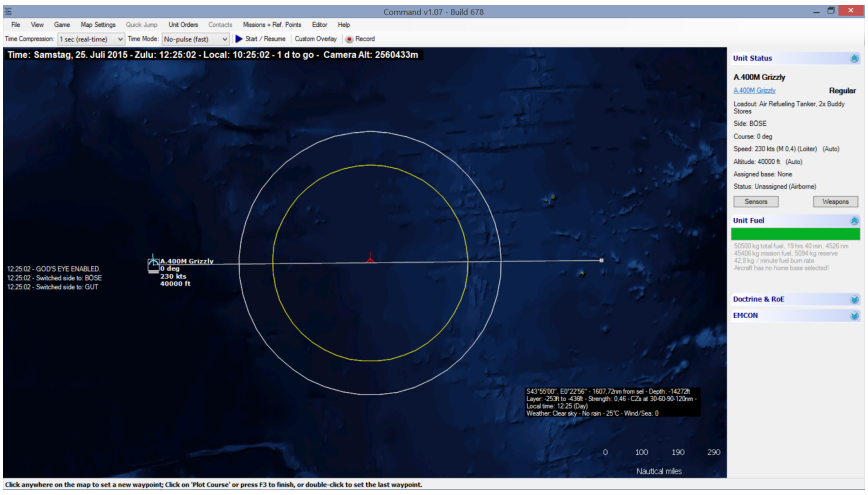

Abbildung 5.8 Szenario 3.

In diesem Szenario habe ich ein AWACS auf der Karte über dem Atlantischen Ozean in einer Höhe von 29.000 ft platziert. Anschließend habe ich einen Airbus A400M aus verschiedenen Höhen anfliegen lassen (siehe Abbildung 5.8). Beginnend nochmals bei 40.000ft und derselben Abfolge, wie in Szenario 2, folgend. Auch hier war das Ziel, ein ähnliches Profil wie im vorherigen Szenario zu erstel-

[3] Genaue Informationen kann man der Website des Herstellers *Boeing* entnehmen, www.boeing.com/history/products/e-3-airborne-warning-and-control-system.page, abgerufen am 20.07.2015.

len und so eine möglichst detaillierte Abbildung des dargestellten Radarbildes zu generieren.

5.4.2 Versuchsablauf

Im ersten Durchlauf habe ich beobachtet, dass das AWACS in einer Höhe von 29.000 ft den Airbus A400M mit einer Flughöhe von 40.000 ft bei nahezu maximaler Reichweite erfasst. Des Weiteren hat das AWACS danach eine durchgehende Radarpeilung des Airbus gehabt, selbst als dieser das AWACS, mit dem nach unten gerichteten Radar, überflogen hat. Demnach ist das AWACS in der Lage auch Flugzeuge zu orten, welche sich oberhalb seiner Flughöhe bewegen.

Abbildung 5.9 AWACS aus dem NATO-AWACS Verband (Bildquelle www.e3a.nato.int/common/images/photo_gallery/features/feature10b.jpg).

In einem weiteren Versuch ließ ich den Airbus bei einer Höhe von 300 ft fliegen, das AWACS weiterhin auf 29.000 ft. Hier beobachtete ich, dass der Airbus circa 100 nautischen Meilen später erfasst wurde.

Die Ergebnisse aus weiteren Testflügen mit unterschiedlichen Höhen konnten mir kein eindeutiges Radarbild liefern, da ich von einem nach unten gerichteten Kegel ausgegangen bin. Aus diesem Grund beschloss ich, alle weiteren Untersuchungen und Beobachtungen zu dem Szenario vorzeitig zu beenden.

5.4.3 Fazit

Meinen ersten Beobachtungen zufolge kann ein AWACS ebenfalls Flugzeuge mit seinem Radar orten, welche sich oberhalb seiner Flughöhe bewegen. Diese Ergebnisse widersprachen meinen bisherigen Kenntnissen von einem gerichteten Radarstrahl. Ebenso verhält es sich mit den übrigen Messwerten. Aus diesen Beobachtungen folgt, dass die Fragestellung durch meine Ergebnisse nur teilweise beantwortet werden kann. Es bleibt demzufolge noch offen zu überprüfen, ob die Radarabbildung eines AWACS, beziehungsweise die Abbildung eines mobilen Radars in der Luft, in dieser Software ebenfalls der Realität nahe kommt.

5.5 Gesamtfazit

Wie zu Beginn dieses Reports genannt, war es Ziel dieser Untersuchungen eine möglichst exakte Vorstellung davon zu erlangen wie *Command: Modern Air/Naval Operations* die Sensoren einzelner Waffensysteme, im speziellen die Radarsysteme, abbildet und wenn möglich zu prüfen, ob dies auch der Realität nahe kommt. Ebenso sollte dieser Bericht meine persönlichen Erfahrungen mit der Software darstellen.

Ich komme im Allgemeinen zu dem Schluss, dass es sich bei dem Spiel um eine Software handelt, welche durchaus das Potential aufweist, zu analytischen Zwecken genutzt zu werden. So konnte ich in verschiedenen Szenarien aufzeigen, dass dieses Spiel physikalischen Grundsätzen folgt, beziehungsweise folgen kann. Dabei ist besonders das Szenario 2 zu nennen, welches beweist, dass sowohl die Erdkrümmung als auch die *Line of Sight* bei der Beeinflussung der Radarortung in der Software mit einbezogen werden. Auch wenn es mir nicht möglich war, jede Frage zu beantworten, hier sei die Rückstrahlenergie von Flugzeugen und die Radardarstellung eines AWACS genannt, so komme ich dessen ungeachtet zu dem Ergebnis, dass es bei dieser Software durchaus im Rahmen des Möglichen liegt, Radarsysteme realitätsnah abzubilden.

In den unbeantworteten Fragen bleibt es dennoch zu prüfen, ob und wie die Spielentwickler bestimmte Gegebenheiten und Szenarien darstellen.

Bei meinen eigenen Erfahrungen im Umgang mit *C:MA/NO* habe ich festgestellt, dass man sich die Handhabung mit dem Spiel sehr schnell und autodidaktisch aneignen kann, wobei die Tutorials sich als sehr nützlich erweisen, und dass es trotz eventueller sprachlicher Hürden oder erster Überforderung dennoch möglich ist sich die spielerischen Fähigkeiten anzueignen. Demnach schätze ich einen Aufwand von 10 bis 15 Stunden, die unerlässlich sind, um den einfachen Umgang zu beherrschen. Die Software ist nichtsdestotrotz derart umfangreich gestaltet, sodass man auch bei einem Aufwand von mehr als 25 Stunden immer noch weitere Aspekte dazu lernt, wobei man bei dieser Software, allem Anschein nach, niemals ausgelernt hat.

Kapitel 6
Piraten am Horn von Afrika

Matthias Schachler

Zusammenfassung Die Software *Command: Modern Air/Naval Operations* bietet eine große Auswahl an Möglichkeiten seine Simulationen aufzubauen. Wie dies geschieht und wie sich die Software dabei verhält werde ich im Folgenden erläutern. Dabei werde ich auf die verschiedenen Einstellungen, die Community sowie die Realitätsnähe anhand von selbsterstellten Untersuchungen eingehen. Diese Untersuchungen beziehen sich auf das Verhalten eines Kampfschiffes in unterschiedlichen Situationen, sowie die dabei gezeigten Möglichkeiten des Handelns.

6.1 Einleitung

Das Magazin „Armchairgeneral“ beschreibt das Spiel *Command: Modern Air/Naval Operations* (kurz: *C:MA/NO*) mit dem Kommentar:

> High levels of realism, massive database, good AI, very high replay value.[1]

Zurzeit sind in *C:MA/NO* nur die im Titel aufgeführten Teilstreitkräfte spielbar, jedoch stehen in der Datenbank auch Teile der Landstreitkräfte zur Verfügung. Diese sind jedoch aufgrund fehlender Detailgenauigkeit noch nicht ohne weiteres Hintergrundwissen spielbar. Es handelt sich um ein umfangreiches Strategiespiel, welches im Detail bereits sehr ausgereift ist, jedoch noch in der Umsetzung verbessert werden könnte. Die Datenbank ist auf den aktuellsten Ständen und wird immer wieder durch die Administratoren aktualisiert. Die Detailgenauigkeit der Waffensysteme ist sehr hoch und wird ebenfalls in den meisten Fällen korrekt dargestellt. Nach einigen Untersuchungen stellte sich jedoch heraus, dass einige Systeme im Unterwasserbereich nicht so funktionieren, wie es in der Realität der Fall ist. Die aktive

Matthias Schachler
Helmut-Schmidt-Universität/Universität der Bundeswehr Hamburg, Holstenhofweg 85, 22043 Hamburg, Deutschland, e-mail: w779821@hsu-hh.de

[1] http://www.armchairgeneral.com/command-modern-air-naval-operations-pc-game-review

Community diskutiert über verschiedene Waffensysteme, die in die Datenbank mit aufgenommen werden sollen, oder verbessert bereits vorhandene Systeme, um diese im Spiel besser darstellen zu können. Diese Community besteht aus mehr als 45.000 Mitgliedern und ist nahezu täglich mit circa 2.000-7.000 Mitgliedern vertreten. Dort sind ungefähr 80 Administratoren aktiv, die die verschiedenen Topics überwachen. Dabei handelt es sich um Fragen und Möglichkeiten zum Spielverhalten selbst, aber auch die Veröffentlichung selbst entwickelter Szenarien durch die Mitglieder. Der Umfang dieser Beiträge beruft sich auf ca. vier bis zehn Beiträge pro Woche und wird je nach Thema durchschnittlich 2.000 bis 11.000 Mal aufgerufen.

Die künstliche Intelligenz (KI) ist in diesem Spiel bereits programmiert, kann jedoch mit möglichen Kenntnissen umprogrammiert werden. Die KI macht jede Mission und jedes Szenario einzigartig, sodass auch bei mehrmaligem Spielen des gleichen Szenarios verschiedene Möglichkeiten offen stehen und das Endergebnis nicht immer identisch ist.

6.2 Einstieg in die Software

Nachdem unter der Leitung unseres Professors eine Einweisung in die Software erfolgte, haben wir diese im Anschluss daran gespielt. Zusätzlich dazu wurde eine Bedienungsanleitung ausgehändigt. Als Spieleinstieg boten sich die „Basic Training"-Missionen an, in welchen man zunächst als aktiver Zuschauer des Geschehens fungierte und keinerlei Mikromanagement an die Einheiten weitergeben musste. Die Missionen waren: Air-, Surface- und Submarine-Operations, welche einen niedrigen Schwierigkeitsgrad und eine geringe Komplexität hatten, um die Grundzüge der Software kennenzulernen. Sie deckten einen wichtigen Teil des Spiels ab, sodass man bereits dort wusste, inwieweit man selbst die Einheiten kontrolliert und Aufträge erstellt, beziehungsweise zuweist. Bereits dabei hat man die Möglichkeiten des Spiels erahnen können, welche sich in den späteren Szenarien als weitaus detaillierter herausstellten, als zuvor vermutet. Auffällig war, dass keine Mission gleich ablief, selbst wenn es das gleiche Szenario war. Dies liegt an der programmierten KI, welche zwar den befohlenen Auftrag verfolgt, jedoch diesen auf unterschiedliche Weise ausführt. Dies wurde bemerkbar, als der gesamte ISA-Kurs die gleiche Basic Training Mission spielte, jedoch Unterschiede bei den einzelnen Checkpoints und dem Ergebnis auftraten.

6.3 Erstes Szenario - Operation Lion's Den 1972

Nachdem die Grundfertigkeiten in den ersten Spielstunden grob erklärt wurden, sollte als nächster Schritt ein bereits fertiggestelltes Szenario selbst gespielt werden. Dies war in meinem Fall die Operation Lion's Den, welche eine Seeschlacht im August 1972 simulierte. Dabei standen sich vier amerikanische Kriegsschiffe,

sowie zwei amerikanische Flugzeuge, den drei feindlichen Torpedoschiffen der Vietnamesen im Hafen von Haiphong gegenüber.[2]

6.3.1 Auftrag

Der Auftrag bestand darin, die Verteidigungslinien rund um das Hafengebiet zu durchbrechen, um anschließend den Hafen selbst, sowie die umliegenden Lagerhallen zu zerstören. Geschichtlich kamen nachdem die amerikanischen Kräfte den Hafen zerstört hatten, die drei vietnamesischen Torpedoboote auf diese amerikanischen Schiffe zu. Wie die Geschichte zeigt, gewannen die amerikanischen Kriegsschiffe gegen die feindlichen Kräfte und erfüllten den Auftrag erfolgreich.

6.3.2 Umsetzung des Szenarios im Spiel

Das Szenario im Spiel war geographisch gut nachgestellt. Allerdings waren die Grenzverläufe dabei grafisch nicht sehr gut angelegt. Die amerikanischen, sowie die vietnamesischen Kräfte waren korrekt im Szenario eingefügt. Aufällig war jedoch, dass der zeitliche Ablauf der Mission nicht mit den Daten der Geschichte übereinstimmte, da teilweise die Torpedoboote bereits auftauchten, als die Mission begann. Des Weiteren stellte sich heraus, dass es keinerlei Punkte für erreichte Missionsziele gab und deshalb die Mission nur mit null Punkten absolviert werden konnte, was anfangs für Verwirrung sorgte. Weiterhin fiel auf, dass durch das frühzeitige Erscheinen der Torpedoboote die Munition sehr schnell verschossen war, sodass keine Munition für den Hafen sowie die Verteidigung an den Küsten zur Verfügung stand. Somit endete das Szenario oftmals so, dass die amerikanischen Truppen vor dem Hafen standen, jedoch keine Munition mehr besaßen, um diesen zu zerstören. Einige Male gelang es den Auftrag geschichtlich korrekt zu erfüllen. Nachdem ich das Szenario selbst verändert habe, sodass die Torpedoschiffe erst später eintrafen, konnte die Mission abgeschlossen werden. Man hat dieses Szenario, egal mit welchem Ende, mit null Punkten abgeschlossen, da der Ersteller des Szenarios keine Ereignisse hinterlegt hat, für welche man Punkte bekommen konnte.

6.4 Eigenes Szenario: Horn von Afrika

Das Szenario beschäftigt sich mit einem aktuellen Thema, dem sog. „swarming“, was bedeutet, dass ein Ziel von mehreren Feinden gleichzeitig bedrängt wird, um

[2] vgl. http://www.uss-newport-news.com/hist/lion's.htm / http://usnavymuseum.org/Ex9_LionsDen.asp

die Chancen des Erfolgs zu erhöhen und das Ziel unter Druck zu setzen. Die Gegner greifen dabei zusätzlich aus mehreren Richtungen an.

6.4.1 Auftrag

Ich habe ein Szenario entwickelt, welches sich mit dem Schussverhalten einer Fregatte bei einer Mehrfachzielbekämpfung auseinandersetzt. Dabei wurde ein Überfall auf ein Kreuzfahrtschiff simuliert, welches von mehreren Schnellbooten im Raum des Horns von Afrika bedroht wird. Die Fregatte wird per Notruf alarmiert und steuert auf die Koordinaten des Kreuzfahrtschiffes zu. Als die Piraten die Fregatte bemerken, eröffnen diese das Feuer und die Fregatte reagiert daraufhin mit der Selbstverteidigung.

6.4.2 Umsetzung des Szenarios im Spiel

Ich habe dem Szenario einen geringen Schwierigkeitsgrad, sowie eine mittlere Komplexität gegeben, da weniger der Auftrag, sondern das Verhalten der Fregatte bei Mehrfachzielbekämpfung untersucht wurde. Dazu wurden drei Parteien gebildet die unterschiedliche Beziehungen zueinander hatten (siehe Tabelle 6.1). Der Schwierigkeitsgrad lässt sich bei der Programmierung durch die eigene Einschätzung des Erstellers eingeben.

Tabelle 6.1 Programmierte Beziehungen zwischen den Parteien

	Fregatte	Piraten	Kreuzfahrtschiff
Fregatte	-	feindlich	freundlich
Piraten	feindlich	-	feindlich
Kreuzfahrtschiff	freundlich	feindlich	-

Nach der Auffassung der feindlichen Kräfte durch den Radar der Fregatte, wurde eine bereits programmierte Verteidigungsmission eingeleitet. Diese war so eingestellt, dass die Ziele erst bekämpft werden sollten, wenn diese das Feuer auf die Fregatte eröffnen. Nachdem die Piraten das Feuer eröffneten und im Erfassungsbereich der Fregatte waren, schoss diese mit den zuvor eingestellten Flugkörpern auf die feindlichen Kräfte. Dabei war zu beobachten, dass Ziele in unmittelbarer Nähe früher bekämpft wurden als andere Ziele. Des Weiteren war zu sehen, dass die gleichzeitige Mehrfachzielbekämpfung in diesem Spiel bereits hinterlegt war. Somit wurden auch bei mehreren Durchläufen des Szenarios immer mehrere Ziele unter Beschuss genommen. Auffällig war allerdings, dass bei einigen Durchläufen die Fregatte den

Rückzug angetreten hat, da die ersten Treffer der feindlichen Kräfte so kritisch waren, dass der Kampf nicht mehr ohne weitere Schäden fortgeführt werden konnte.

6.4.3 Schadensanzeige

Ein weiterer Aspekt, welcher untersucht wurde, waren die Schadensanzeigen. Dabei wird in vier Kategorien unterschieden:

- Light damage
- Medium damage
- Heavy damage
- Destroyed

Die Beschädigungen wirken sich dabei auf die Kampfbereitschaft des jeweiligen Ziels aus. Dies hat zur Folge, dass beispielsweise ein Schiff mit steigendem Schaden, nur eine begrenzte Geschwindigkeit fahren kann, oder einzelne Systeme nicht mehr funktionieren. Neben der Schadensanzeige zeigt die Simulation auch Feuer-

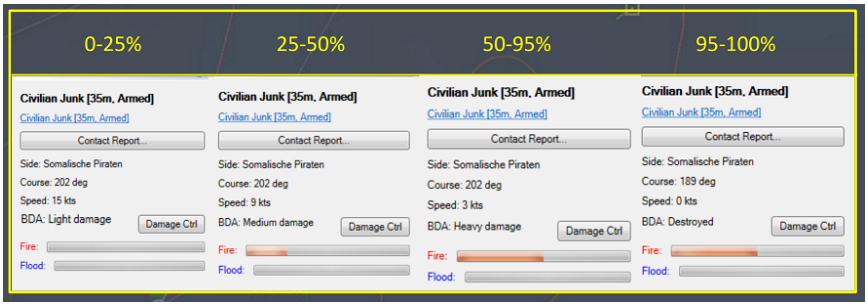

Abbildung 6.1 Schadensanzeige

und Flutstand am eigenen Objekt an. Diese veranschaulichen, inwieweit die jeweiligen Faktoren am Objekt vorhanden sind. Je höher die jeweilige Anzeige ausschlägt, desto schwerwiegender ist der daraus erlittene Schaden für das eigene Schiff. In der Simulation dauert es länger, ein großes Feuer zu löschen, als ein kleines. Gleiches gilt beim Wassereintritt. Das Löschen eines Feuers oder das Verhindern eines Wassereintritts wird automatisch vom Spiel eingeleitet. Lediglich die Dauer ist je nach Größe variabel. Die Schadensanzeige steigt so lange proportional zur Feuer-/Flutanzeige, bis die Ursache beseitigt wurde. Das heißt, der erlittene Schaden steigt schneller, wenn die Anzeige zu 100 Prozent ausgelastet ist und langsamer, wenn die Anzeige nur zu 25 Prozent gefüllt ist. Sind Wassereintritt oder Feuer beseitigt, so verbleibt die Schadensanzeige an ihrem letzten Stand. Dieser Prozentsatz ist dann je nach Höhe in einer der vier Kategorien aus Abbildung 6.1 einzuordnen.

6.4.4 Waffenwirkungen im Szenario

Im Szenario traten mehrere Arten der Waffenwirkung auf. Die Raketen trafen nicht immer ihr Ziel, oder beschädigten das Ziel nur zu einem bestimmten Prozentsatz (siehe Abbildung 6.1). Das Spiel unterscheidet in vier Arten bei dem im Szenario verwendeten Waffensystem:

- Missile is running blind for more than 6 sec ... self-destructing.
- Missile airbursted directly on top of [Target]
- Missile malfunctioned
- Missile airbursted off [Target] by 7ft

Dass Blindgänger sowie Fehlfunktionen im Spiel mit aufgenommen wurden ist durchaus positiv zu bewerten. Dies entspricht der Realität und geschieht im Spiel zu einem bestimmten Prozentsatz (siehe Abbildung 6.2).

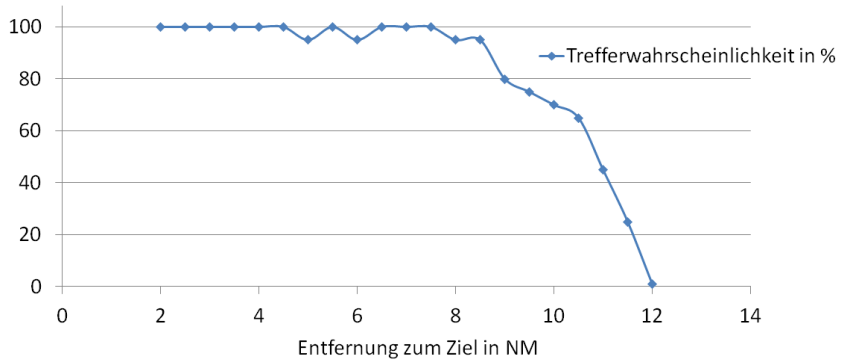

Abbildung 6.2 Trefferwahrscheinlichkeiten von Lenkflugkörpern

6.4.5 Verhalten von Flugzeugen bei Kraftstoffmangel

Die Simulation ist sehr realistisch dargestellt, wenn es um Verbrauch von Treibstoff in verschiedenen Höhen, sowie Geschwindigkeiten geht. Dabei die Daten des Flugzeugs aus Abbildung 6.3 angezeigt.

Dabei startet der Eurofighter in diesem Fall mit 5.000kg Treibstoff, wobei er beim Landen noch über einen Rest von 800kg (grob 20NM bei Normalflug) im Tank verfügt.

Bei der Wahl der Homebase entscheidet die KI automatisch welche Homebase gewählt wird. Lässt man das Flugzeug zwischen zwei Basen in der Simulation star-

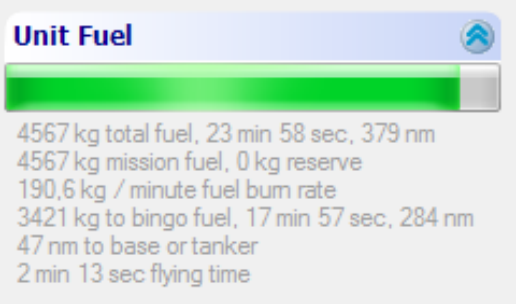

Abbildung 6.3 Treibstoffanzeige

ten, so wählt die KI die Basis aus, welche als erstes errichtet wurde, auch wenn das Flugzeug näher an einer anderen Basis liegt. Zu dieser ausgewählten Basis kehrt das Flugzeug auch immer dann zurück, wenn der Treibstoff nur noch für den Rückweg reicht. Man kann diese Einstellungen jedoch manuell ändern. Liegt eine andere Basis taktisch besser am Einsatzgebiet, so kann man der Einheit verschiedene Befehle während der Mission zuweisen:

- Return to Base
- Select new home Base
- Refuel if possible

Nur bei „Select new home Base“ fliegt das Flugzeug zu der neu zugeteilten Basis. Bei den anderen beiden Punkten fliegt es zur erst errichteten Basis zurück. Ausweichen kann man den Vorgaben des Spiels, indem man ein Tankflugzeug in einem Bereich einsetzt und das Flugzeug diesem Tanker zuweist. Auch dann berechnet die KI selbstständig, wann das Flugzeug zum Tanker fliegen muss, um noch genug Treibstoff als Reserve zu haben. Befehlt man dem Flugzeug trotz mangelnden Treibstoffs weiterzufliegen, so wird es den *Point of no return*[3] überfliegen und aufgrund des Treibstoffmangels abstürzen. So ist es in der Simulation, aber auch in der Realität. Es ist dabei nicht möglich, das Flugzeug notlanden zu lassen, wenn keine angepasste Basis vorhanden ist. Dieser Punkt ist speziell bei Ozeanüberflügen, aber auch auf dem Festland wichtig, jedoch in der Simulation nicht durchführbar. Die Weltkugel in der Simulation dient nur als örtlicher Orientierungspunkt, die KI erkennt nur erstellte Objekte, nicht aber die einzelnen Details auf dem globalen Abschnitt, welche als Notlandeplätze dienen könnten.

[3] Als Point of no return bezeichnet man den Punkt, bei welchem der verbleibende Treibstoff nicht mehr für den Rückflug zur Ausgangsbasis ausreicht und bis zur nächsten Basis weitergeflogen werden muss.

6.5 Zusammenfassung

Zusammenfassend lässt sich die anfangs gestellte Leitfrage folgendermaßen beantworten:

> **Bei *Command: Modern Air/Naval Operations* handelt es sich sowohl um ein Strategiespiel, als auch um eine Simulationssoftware.**

Beide Aspekte sind in gleichem Masse vertreten wobei man dazu sagen muss, dass die Software sich nicht als Analysesoftware eignet, da das Spielen im Vordergrund steht. Das heißt, es ist nicht möglich, innerhalb kürzester Zeit ein Szenario 1000 Mal durchlaufen zu lassen, um Daten zu vergleichen, da man den Ablauf jedes Mal erneut starten muss und die maximale Spielgeschwindigkeit bei 30min/sek liegt. Dies kann bei komplexen Simulationen ein Hindernis sein. Abschließend sind die positiven und negativen Aspekte von *C:MA/NO* in Abbildung 6.4 zusammengefasst.

Positiv	Negativ
Aktuelle Datenbank mit Objekten aller Art	Grenzverläufe sind auf der Karte teilweise ungenau
Viele Szenarien sind abbildbar	Kann nicht als Analysesoftware genutzt werden
Aktive Community	Trotz guter Datenbank ist Umsetzung der Waffen ungenau
Viele Einstellungen möglich	Szenarien werden irgendwann unübersichtlich
Viel Kontrolle	Wettereinflüsse werden nicht mit berücksichtigt
Randomfaktor bringt Abwechslung	Leistungsstarker PC notwendig, bei komplexen Szenarien
Günstige Anschaffungskosten	
Detaillierte geographische Angaben	
Intelligente KI	

Abbildung 6.4 Positive und negative Aspekte von *C:MA/NO*

Kapitel 7
Fazit

Die Kursteilnehmer haben sich während des Trimesters eingehend mit der Simulationsumgebung *C:MA/NO* beschäftigt. Neben den wöchentlichen Sitzungen hat jeder Einzelne viel Freizeit investiert und war nach eigener Auskunft mit Spaß bei der Sache. Das Programm ist somit in der Lage, seine Benutzer nachhaltig zu motivieren. Ein weiterer positiver Nebeneffekt liegt im Erlernen der englischen Sprache (insbesondere des militärischen Fachvokabulars), was für die erfolgreiche Benutzung unabdingbar ist, und hier mit spielerischer Leichtigkeit nebenher erfolgt.

Die Lernkurve ist anfangs recht steil. Die Vielzahl der Menüs, Untermenüs und umfangreichen Datenbanken erschlägt den Erstnutzer anfangs. Das dicke Handbuch bestärkt die Zweifel eher, als dass es sie nimmt. Jedoch stellte jeder fest, dass man nach wenigen Stunden bereits in der Lage war, die mitgelieferten Szenarien durchzuspielen, und nach wenigen weiteren Stunden auch erfolgreichen abzuschließen. Nach etwa 20 Stunden konnten Szenarien selbst kreiert werden, wobei der Phantasie und Detailgenauigkeit (und damit der weiter investierten Zeit) keine Grenzen gesetzt sind.

Im eigentlichen Kern der Sache konnte *C:MA/NO*, wie der Programmtitel es schon andeutet, insbesondere bei Luft- und Seemanövern überzeugen. Viele physikalische und technische Phänomene sind in der Simulation erfasst und werden korrekt wiedergegeben, wenngleich mit vereinfachten Modellen. Die künstliche Intelligenz zur Steuerung der eigenen oder gegnerischer Entitäten trifft überwiegend nachvollziehbare Entscheidungen. Jedoch konnte jeder Student im Rahmen seiner Arbeiten auch Schwächen der Modelle aufspüren, so dass man die Resultate nicht unkritisch und ungeprüft verwenden sollte. Die größten Probleme bekommt *C:MA/NO* dann, wenn man es für Landoperationen „missbraucht“.

In Anbetracht der schnellen Einarbeitungszeit und der überwiegend korrekten Darstellung und Verhaltens der Systeme kann *C:MA/NO* empfohlen werden, wenn mit geringem Aufwand an Geld, Zeit und Personal ein größtmöglicher Simulationsnutzen erzielt werden soll. Für weitergehende, verfeinerte Analysen (die entsprechend mehr Ressourcen benötigen) wird man im Anschluss auf andere Simulationsumgebungen zurückgreifen müssen.