

FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO



FEUP

Interface de Operação para Veículos não Tripulados

António Sérgio Ferreira

Mestrado Integrado em Engenharia Informática e Computação

Orientador: Gil Manuel Gonçalves (Eng.)

24 de Fevereiro de 2011

Interface de Operação para Veículos não Tripulados

António Sérgio Ferreira

Mestrado Integrado em Engenharia Informática e Computação

Aprovado em provas públicas pelo júri:

Presidente: Raúl Moreira Vidal (Professor)

Vogal Externo: César Analide (Professor)

Orientador: Gil Manuel Gonçalves (Engenheiro)

10 de Fevereiro de 2011

Resumo

Cada vez mais são evidentes os benefícios da utilização de equipas robóticas na execução de actividades demasiado exaustivas ou perigosas para seres humanos. Um excelente exemplo disso pode ser encontrado nas Forças Armadas, onde a utilização de uma equipa deste género pode poupar a vida a vários soldados, ao mesmo tempo que permite cumprir missões de extensa duração, onde o sucesso poderia ser posto em causa devido à natural exaustão humana. No entanto, apesar dos ganhos logísticos e em recursos humanos, é necessário permitir que uma equipa deste género seja facilmente controlada e supervisionada por um elemento humano que consiga garantir a mais valia estratégica deste novo recurso.

Esta dissertação nasce com a directriz de criar um módulo de operações central que permita a supervisão de uma equipa de veículos aéreos não-tripulados por apenas um elemento humano, que orientará a equipa e fornecerá os ajustes necessários ao sucesso de cada missão. A implementação deste módulo auxilia-se em conceitos base dos videojogos de estratégia em tempo real. Com isto, tenta-se transportar as lições aprendidas ao longo de décadas, referentes a formas eficientes de gerir e monitorizar grupos de média a grande escala, para o novo módulo de operações.

Após a recolha de dados, relativos ao uso de videojogos de estratégia como base de concepção nesta área, foi efectuada uma análise das métricas que seriam usadas para avaliar a performance do supervisor e a qualidade do módulo de operações desenvolvido. Feito este estudo, foi possível partir para uma visão mais detalhada da *framework* base do projecto - o Neptus - de forma a compreender a sua estrutura, assim como o ponto de partida para a implementação do módulo. No final da implementação do módulo, foram efectuados simulacros de voo com diferentes tamanhos de equipas robóticas, com o objectivo de verificar a robustez e performance da nova ferramenta.

Desta forma, o produto final desta dissertação é um módulo de operações que permite a supervisão de uma equipa de veículos aéreos não tripulados, tendo os resultados obtidos nos simulacros provado a existência de um aumento na performance do supervisor humano, graças às modificações implementadas na concepção da ferramenta.

Palavras-Chave: HRT; UAV's; HCI; Jogos de Estratégia em Tempo Real; Supervisão

Abstract

Increasingly evident are the benefits of using robotics teams in activities too exhausting or dangerous for humans. An excellent example can be found in the military where the use of a team of this kind can save the lives of several soldiers, while allowing the successful completion of missions of extreme duration which could be jeopardized by natural human exhaustion. However, despite the gains in logistics and human resources it is crucial to provide an easy and simple way of monitoring and supervising these teams by a human element, that can ensure this new feature is used to its fullest potential.

This dissertation is launched with the guideline to create a central operation module that allows the supervision of a team of unmanned aerial vehicles by only one human element, that element will guide the team and provide the necessary adjustments to the success of each mission. The implementation of this module gets its basis from real-time strategy video games concepts. By this we mean to transport the lessons learned over decades, related to efficient ways of managing and monitoring groups of medium to large scale, from this medium to the new module of operations.

After gathering data on the use of real-time strategy video games in this area there was an examination metrics that would be used to evaluate the performance of the supervisor and the quality of the developed operation module. With the completion of this study it was then possible to obtain a more detailed view of the base framework for the project - Neptus - in order to understand its structure and what would be the starting point for implementing the module. At the end of the implementation phase there were made flight simulations with different size robot teams in order to verify the robustness and performance of the new tool.

Thus, the final product of this work is an operations module that enables the monitoring of a team of unmanned aerial vehicles. The results obtained from simulations proved that there was an increase in the human supervisor performance thanks to changes implemented in the design tool.

Keywords: HRT; UAV's; HCI; Real Time Strategy Game; Supervision

Agradecimentos

Gostaria de começar por estender os meus agradecimentos ao Professor Gil Gonçalves, pela forma como contribuiu para este documento e todo o projecto, mostrando sempre disponibilidade e preocupação para que esta dissertação chegasse a bom porto.

Seguidamente, gostaria de agradecer a todos os elementos do Neptus e AsasF, pois graças à paciência e boa vontade de todos, foi possível completar este projecto e atingir os resultados propostos.

Da mesma forma, quero agradecer à Faculdade de Engenharia da Universidade do Porto por ter sido, durante estes quase seis anos, a minha segunda casa, para grande preocupação dos meus pais. Foi o único sítio onde realmente me consegui concentrar no trabalho e foi também aqui que conheci os meus melhores amigos.

E é agora para eles que me dirijo quando digo obrigado por todos os momentos passados, desde os risos às discussões, desde as noites mais trabalhosas às festas mais relaxantes e desde o Windows ser melhor que o Linux e o Mac também. Vocês deram vida a uma experiência que poderia ter sido, de outra forma, muito enfadonha e desinteressante.

Antes de terminar é necessário, igualmente, agradecer à Dra. Rita Duro por toda a paciência durante este projecto, pela dedicação e determinação que me fizeram manter o rumo. Como uma rocha, sempre pude contar com ela, especialmente quando os nervos se apoderavam de mim. Sem ela, este documento estaria muito diferente.

Finalmente, aos meus pais e irmão, por terem acreditado que este rapaz despassarado alguma vez conseguiria ser alguém decente. Agradeço especialmente pela paciência e carinho nos momentos mais difíceis.

António Sérgio Ferreira

“Brevity is the soul of wit”

From Shakespeare’s Hamlet, 1602

Conteúdo

1	Introdução	1
1.1	Contexto/Enquadramento	1
1.1.1	Projecto PITVANT	2
1.2	Motivação e Objectivos	2
1.3	Estrutura da Dissertação	4
2	Revisão Bibliográfica	5
2.1	Equipas Robóticas	5
2.1.1	Homogéneas	5
2.1.2	Heterogéneas	6
2.2	Sistemas Correntes	7
2.2.1	Macbeth	7
2.2.2	MOCU	8
2.3	Paradigma de Jogos RTS	9
2.3.1	Modos de Representação	10
2.3.2	Modo de Interacção	13
2.4	Métricas de Avaliação	14
2.4.1	Fan-Out	14
2.4.2	Situation Awareness	16
2.4.3	Workload	18
2.5	Resumo	18
3	Análise da Solução em Neptus	19
3.1	Neptus	19
3.1.1	Introdução	19
3.1.2	Arquitectura	20
3.2	IMC	22
3.3	Dune	23
3.4	Piccolo	23
3.5	Ambiente entre o Software de Missão	24
3.6	Metodologia de Operações com UAV's	25
3.7	Requisitos do Problema	28
3.7.1	User Story	28
3.7.2	Requisitos Funcionais	29
3.7.3	Requisitos Não Funcionais	32
3.8	Base de Desenvolvimento	32
3.9	Resumo	33

CONTEÚDO

4	Implementação	34
4.1	Arquitectura do plugin UAV	34
4.1.1	Panels	34
4.1.2	Painters	35
4.1.3	Listeners	35
4.2	Arquitectura do Módulo de Operação	37
4.2.1	Uav Manager Panel	37
4.2.2	Uav Mini Map Panel	39
4.2.3	Uav Altitude Panel	40
4.2.4	Uav Status Panel	40
4.2.5	Uav Function Panel	42
4.2.6	System List	43
4.2.7	Image Evaluation Panel	43
4.2.8	Generic Displays	45
4.3	Cumprimento de Requisitos Funcionais	46
4.4	Resumo	47
5	Resultados	48
5.1	Testes Efectuados	48
5.1.1	Workload	48
5.1.2	Situation Awareness	51
5.1.3	Fan-Out	51
5.1.4	Curva de Aprendizagem	56
5.2	Cumprimento de Requisitos Não Funcionais	57
5.3	Análise aos Componentes do Módulo	57
5.4	Resumo	58
6	Conclusões e Trabalho Futuro	59
6.1	Resumo do Trabalho Efectuado	59
6.2	Satisfação dos Objectivos	60
6.3	Trabalho Futuro	61
	Referências	63
A	Simulações de Teste	67
A.1	Simulação 01	67
A.1.1	Constituição da Equipa	67
A.1.2	Objectivos	67
A.1.3	Plano de Operação	68
A.1.4	Análise NASA-TLX	69
A.1.5	Análise SAGAT	71
A.1.6	Questionário de Avaliação	72
A.2	Simulação 02	73
A.2.1	Constituição da Equipa	74
A.2.2	Objectivos	74
A.2.3	Plano de Operação	74
A.2.4	Análise NASA-TLX	75
A.2.5	Análise SAGAT	77

CONTEÚDO

A.2.6	Questionário de Avaliação	78
B	Ferramenta NASA-TLX	80
B.1	Determinação do Peso da Componente	80
B.2	Determinação dos Valores Relativos	81
B.3	Determinação dos Valores Pesados	82
C	Ferramenta SAGAT	83
C.1	Modelo	83
C.2	Metodologia	83
D	Manual de Inicialização do Módulo	84
D.1	Preparação Inicial	84
D.2	Piccolo Command Center	84
D.3	Dune	85
D.4	Neptus	86

Lista de Figuras

2.1	Exemplo de uma HRT homogénea	5
2.2	Exemplo de uma HRT heterogénea	6
2.3	Exemplo da interface do sistema Macbeth [SAF ⁺ 00]	7
2.4	Exemplo da interface do sistema MOCU [Pow06]	8
2.5	Exemplo da interface de um jogo RTS	9
2.6	Exemplo da interface desenvolvida em [HSS07], em modo de visão isométrica	11
2.7	Exemplo da interface desenvolvida em [HSS07], em modo de visão na primeira pessoa	11
2.8	Exemplo dos vários tipos de mapas 2D: (a) 2D com rótulos; (b) 2D com elevações - 2,5D; (c) 2D com imagens reais; (d) 2D com imagens e elevações [LRY ⁺ 07]	12
2.9	Fórmula simplificada da métrica <i>fan-out</i> [OW04]	14
2.10	Limite atingido de <i>fan-out</i> [OW04]	15
2.11	Variante do calculo para o <i>fan-out</i> [WL08]	15
2.12	Extensão do conceito de <i>neglect time</i> [WL08]	16
2.13	Exemplo de ícons utilizados em [KWSN06] para a investigação dos benefícios da abordagem multi-modal	16
2.14	Resultados obtidos referentes aos benefícios da abordagem multi-modal [KWSN06]	17
3.1	Exemplo da interface do módulo Neptus	20
3.2	Arquitectura simplificada do código-fonte do Neptus	21
3.3	Exemplo da constituição das consolas Neptus	21
3.4	Diagrama exemplo de mensagens IMC [MDM ⁺ 09]	22
3.5	Imagem de um exemplar Piccolo [pic]	24
3.6	Esquema a actual interacção entre o software necessário para o voo de um UAV	25
3.7	Esquema da desejada interacção entre o software necessário para o voo de um UAV	26
3.8	Esquematisação dos elementos participantes numa missão de UAV	27
3.9	Diagrama de Casos de Uso do módulo desenvolvido	29
3.10	Exemplo da interface principal da consola a desenvolver	30
3.11	Exemplo do funcionamento da galeria de fotos recolhidas pelos UAV's	31
3.12	Representação dos elementos aproveitados, ou alterados, na implementação do módulo de operação	33
4.1	Esquema da arquitectura do <i>plugin</i> desenvolvido para o Neptus	35

LISTA DE FIGURAS

4.2	Diagrama de classes do <i>plugin</i> UAV	36
4.3	Esquema da arquitectura do módulo de operações desenvolvido	37
4.4	Esquema da disposição dos componentes do módulo de operações	38
4.5	Protótipo do painel de supervisão principal	38
4.6	Protótipo do mini mapa usado em conjunção com o painel de supervisão principal	40
4.7	Protótipo do painel utilizado para monitorizar a altitude dos UAV's	41
4.8	Protótipo do painel representativo do estado das várias secções do UAV	42
4.9	Protótipo do painel multiusos da consola	42
4.10	Imagem do painel Neptus que lista os veículos activos numa consola	43
4.11	Protótipo do painel de avaliação de imagens	44
4.12	Exemplo do uso de cores para transmitir informação	45
4.13	Exemplo de formato de ícon para transmitir informação	45
4.14	Imagem exemplo da utilização de <i>displays</i> genéricos existentes no Neptus	46
5.1	Gráfico dos diferentes pesos atribuídos às componentes da ferramenta NASA-TLX no Simulacro 01	49
5.2	Gráfico dos diferentes pesos atribuídos às componentes da ferramenta NASA-TLX no Simulacro 02	49
5.3	Comparação do <i>workload</i> total entre os diferentes módulos de operação no Simulacro 01	50
5.4	Comparação do <i>workload</i> total entre os diferentes módulos de operação no Simulacro 02	51
5.5	Gráfico apresentando a comparação do <i>workload</i> , dividido nas suas componentes, entre os diferentes módulos de operação no Simulacro 01	52
5.6	Gráfico apresentando a comparação do <i>workload</i> , dividido nas suas componentes, entre os diferentes módulos de operação no Simulacro 02	52
5.7	Apresentação da variação percentual do <i>workload</i> registada para cada componente no Simulacro 01	53
5.8	Apresentação da variação percentual do <i>workload</i> registada para cada componente no Simulacro 02	53
5.9	Representação das diferenças de SA entre ambos os módulos no Simulacro 01	54
5.10	Representação das diferenças de SA entre ambos os módulos no Simulacro 02	54
5.11	Gráfico da variação percentual do SA entre ambos os módulos no Simulacro 01	55
5.12	Gráfico da variação percentual do SA entre ambos os módulos no Simulacro 02	55
A.1	Representação das áreas de vigilância do Simulacro 01	68
A.2	Representação das áreas de vigilância do Simulacro 02	75
B.1	Exemplo dos pares de avaliação de pesos para cada componente do <i>workload</i>	80
B.2	Exemplo da escala usada para recolher os valores de <i>workload</i> para cada componente	81

LISTA DE FIGURAS

D.1	Verificação da comunicação PCC	85
D.2	Interface inicial do PCC	85
D.3	Exemplo da interface do módulo Neptus	86
D.4	Interface inicial do Neptus	87
D.5	Exemplo da interface do módulo Neptus	87
D.6	Exemplo da interface do módulo Neptus	87

Lista de Tabelas

3.1	Lista de prioridades dos requisitos funcionais	31
3.2	Lista de prioridades dos requisitos não funcionais	32
4.1	Tabela de cumprimento dos requisitos funcionais	46
5.1	Tabela de cumprimento dos requisitos não funcionais	57
A.1	Tabela de distribuição de pesos entre componentes de <i>workload</i>	70
A.2	Tabela de distribuição do <i>workload</i> , não pesado, entre componentes, no módulo antigo	70
A.3	Tabela de distribuição do <i>workload</i> , não pesado, entre componentes, no módulo novo	70
A.4	Tabela de distribuição do <i>workload</i> , pesado, entre componentes, no módulo antigo	71
A.5	Tabela de distribuição do <i>workload</i> pesado, entre componentes, no módulo novo	71
A.6	Tabela de distribuição da SA no módulo antigo	71
A.7	Tabela de distribuição da SA no módulo novo	72
A.8	Tabela de distribuição de pesos entre componentes de <i>workload</i>	76
A.9	Tabela de distribuição do <i>workload</i> , não pesado, entre componentes, no módulo antigo	76
A.10	Tabela de distribuição do <i>workload</i> , não pesado, entre componentes, no módulo novo	76
A.11	Tabela de distribuição do <i>workload</i> , pesado, entre componentes, no módulo antigo	77
A.12	Tabela de distribuição do <i>workload</i> , pesado, entre componentes, no módulo novo	77
A.13	Tabela de distribuição da SA no módulo antigo	77
A.14	Tabela de distribuição da SA no módulo novo	78

Abreviaturas e Símbolos

AGL	Above Ground Level
AT	Activity Time
API	Application Programming Interface
CSCW	Computer Supported Collaborative Work
FEUP	Faculdade de Engenharia da Universidade do Porto
FO	Fan-Out
FT	Free Time
GS	Ground Station
GPS	Global Positioning System
HRI	Human Robot Interaction
HRT	Human Robot Team
IMC	Inter-Module Communication
IT	Interaction Time
LSTS	Laboratório de Sistemas e Tecnologia Subaquática
MOCU	Multi-Robot Operator Control Unit
NASA	National Aeronautics and Space Administration
NASA-TLX	NASA Task Load Index
NT	Neglect Time
SAGAT	Situation Awareness Global Assessment Technique
SART	Situation Awareness Rating Technique
UAV	Unmanned Air Vehicle
VoIP	Voice over Internet Protocol
OT	Occupied Time
PITVANT	Projecto de Investigação e Tecnologia em Veículos Aéreos Não - Tripulados
PC	Personal Computer
PCC	Piccolo Command Center
XML	Extensible Markup Language

Capítulo 1

Introdução

Com o expandir do poderio tecnológico a nível da robótica e do software, tem sido possível, cada vez mais, utilizar *human robot teams* (HRT) para a execução de operações tipicamente consideradas de risco. Operações de reconhecimento, salvamento, [MWKS06] [YDS04] e até mesmo de origem militar [Pow06] são agora possíveis de levar a cabo de forma a minimizar o perigo para o ser humano. No entanto, esta descentralização de perigo coloca um peso acrescido na necessidade de uma boa performance por parte da HRT [PJS⁺10] [CC07], especialmente a nível da supervisão no decorrer das operações. É com a intenção de maximizar a performance e eficácia do supervisor da HRT que esta dissertação é proposta. Para cumprir essa premissa, será efectuado um estudo às áreas de supervisão e controlo de HRT, com o objectivo de projectar e implementar um protótipo de um módulo de operações que, através da incorporação de noções existentes em *real-time strategy games* (RTS), irá permitir um melhor desempenho do supervisor responsável pela HRT.

1.1 Contexto/Enquadramento

Esta dissertação enquadra-se num projecto de maior envergadura, o projecto PIT-VANT (Projecto de Investigação e Tecnologia em Veículos Aéreos Não - Tripulados), lançado pela Academia da Força Aérea, em parceria com a FEUP (Faculdade de Engenharia da Universidade do Porto). A implementação do módulo de operações, que será desenvolvido no decorrer da dissertação, irá efectuar-se dentro do projecto Neptus [PDG⁺06] [DGP⁺07], à guarda do Laboratório de Sistemas e Tecnologia Subaquática (LSTS), sediado na FEUP. Todo o desenvolvimento será acompanhado pela equipa de desenvolvimento do Neptus, de forma a possibilitar a inserção segura deste módulo na

arquitetura actual da ferramenta, e será igualmente feito em parceria com a equipa responsável pela esquadilha de UAV's do LSTS, os AsasF. Com o input da equipa AsasF, podemos determinar melhor as necessidades operacionais do supervisor no decorrer de uma missão, garantindo assim que o novo módulo de operações é desenvolvido tendo constantemente em vista a sua utilização prática. De igual forma, os AsasF serão responsáveis pelos testes que permitirão determinar a aceitação do módulo.

De forma a melhor compreender a escala do projecto, é feita uma introdução ao programa PITVANT.

1.1.1 Projecto PITVANT

O projecto PITVANT surge como a continuação do projecto desenvolvido na Academia da Força Aérea em 1996, com o objectivo de lhe fornecer uma esquadilha de UAV's, para ir dotando a Força Aérea de capacidades, cada vez mais indispensáveis, de exploração e vigilância, quer com objectivos militares, quer com objectivos civis.

O PITVANT está dividido em várias fases, tendo como objectivos essenciais:

- Desenvolver protótipos de sistemas de UAV's, essencialmente de pequena e média dimensão, integrando as mais modernas tecnologias, a usar em diversas missões e actividades, não só de natureza estritamente militar, mas também de natureza civil, incluindo, naturalmente, actividades de investigação;
- Fornecer à Força Aérea *know-how* relativo à definição de requisitos técnicos e operacionais, à utilização e à operação de UAV's;
- Criação de um projecto de I&T em conjunto com a FEUP, mais concretamente com o LSTS, para a criação e desenvolvimento de hardware e software para os UAV's.

Esta três fases começaram a ser desenvolvidas em Novembro de 2008 e prolongar-se-ão até Dezembro de 2015.

No decorrer do restante documento, são explicitadas as dificuldades inerentes à elaboração de um sistema deste género, assim como a forma escolhida para alcançar a sua solução.

1.2 Motivação e Objectivos

A um nível prático, o trabalho desenvolvido irá possibilitar libertar recursos humanos de tarefas de gestão e supervisão, em cenários que protagonizem HRT's de grande envergadura. De igual modo, será possível reduzir custos logísticos, associados à possível necessidade de existir hardware específico à tarefa de monitorização da HRT. No entanto,

é necessário referir que esta centralização de funções e responsabilidade num só posto vem acompanhada de complicações que, não sendo abordadas, poderão colocar em causa a viabilidade do projecto. Não obstante, as complicações desta concentração são reconhecidas e serão enfrentadas no decorrer da dissertação, através da constante vigilância, recorrendo a prototipagens iterativas.

Estes protótipos irão permitir:

- Identificar requisitos ambíguos ou que estejam em falta, com o intuito de construir um módulo o mais fiel possível à experiência e necessidade de supervisores reais;
- Recolher feedback constante da equipa responsável pelos testes ao módulo, e, em última instância, os seus clientes finais, os AsasF.

Adicionalmente, e considerando que o projecto está a ser desenvolvido dentro do Neptune - ferramenta já largamente utilizada dentro do LSTS para controlar HRT's de *unmanned underwater vehicles* (UUV) -, é necessário reconhecer as componentes concebidas e implementadas para o módulo de operações para os UAV's que podem ser expandidas para o cenário aquático dos UUV's, aumentando assim a utilidade do trabalho desenvolvido durante a dissertação.

A um nível académico, é correcto admitir que, com o desenvolvimento deste projecto, será possível validar, cada vez mais, a utilidade trazida pela indústria dos videojogos ao desenvolvimento de projectos com objectivos científicos.

Como objectivos centrais da dissertação temos:

- O desenvolvimento de um módulo de operações, baseado em princípios de jogos RTS, que permita a um supervisor humano gerir uma HRT de UAV's;
- A maximização do número de robôs possíveis de ser supervisionados, simultaneamente, por esse supervisor humano;
- O aumento da performance do supervisor, comparativamente ao método actual de supervisão, durante uma missão com UAV's;
- A oportunidade de facultar um estudo apoiado em métricas correntes na área de HRT (*fan-out*, *situation awareness*, etc.).

Tomando tudo isto em consideração, será então possível expandir o espectro de missões em que estas equipas poderão ser empregues com sucesso, indo assim de encontro ao crescente desejo de utilizar HRT's como forma de libertar o ser humano de situações de perigo.

1.3 Estrutura da Dissertação

Para além da introdução, esta dissertação contém mais cinco capítulos:

- No capítulo 2, é descrito o estado da arte referente à dissertação. Inicialmente, é feita uma exposição dos tipos de HRT existentes, seguindo-se uma apresentação de sistemas que já permitem efectuar supervisão de HRT. Além disso, é efectuada uma introdução ao paradigma de jogos RTS, tendo em vista clarificar pontos importantes que irão afectar a concepção do módulo. Finalmente, são expostas as métricas a serem utilizadas para a avaliação dos objectivos propostos;
- No capítulo 3, é feita uma apresentação da ferramenta Neptus, bem como a metodologia usada pelos AsasF durante uma missão com UAV's. São, de seguida, expostos os requisitos funcionais e não funcionais do projecto e é feita a apresentação das bases do Neptus que serão usadas na implementação do módulo de operações;
- No capítulo 4, é delineada a arquitectura do módulo de operações desenvolvido, assim como os detalhes dos seus componentes. É então feita uma análise do cumprimento dos requisitos funcionais;
- No capítulo 5, são apresentados os resultados recolhidos durante os testes ao módulo de operações e é feita uma análise a cada métrica utilizada na avaliação. Como nota final, são analisados os requisitos não funcionais, de forma a determinar o seu cumprimento;
- No capítulo 6, é feito um resumo dos passos tomados durante a dissertação, assim como uma análise do trabalho concluído. São igualmente expostos desenvolvimentos futuros, que podem acrescentar valor ao módulo desenvolvido, aumentando assim a sua utilidade e versatilidade.

Capítulo 2

Revisão Bibliográfica

2.1 Equipas Robóticas

Tendo esta dissertação como objectivo fundamental desenvolver um sistema que optimize a supervisão de uma HRT, é necessário começar por definir, de forma clara, os dois tipos de equipas que podem surgir dentro do contexto existente.

2.1.1 Homogéneas

Sendo este o paradigma mais comum para uma HRT, o factor de homogeneidade permite, desde logo, impor algumas restrições no que toca à forma de representar, visualmente, a equipa de robôs. Isto simplifica consideravelmente a construção do módulo de visualização para operações, visto não ser necessário ter em consideração a forma como diferentes representações se irão afectar mutuamente. Adicionalmente, o facto de toda a equipa ser composta por elementos do mesmo modelo (Figura 2.1) e, consequentemente, possuírem as mesmas capacidades e funcionalidades, facilita a generalização e

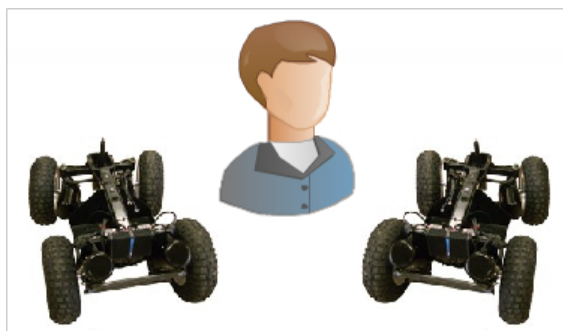


Figura 2.1: Exemplo de uma HRT homogénea



Figura 2.2: Exemplo de uma HRT heterogênea

a extrapolação de conclusões, extraídas de dados recolhidos empiricamente, referentes quer a métricas de performance, quer a heurísticas de construção de interfaces de controlo. Isto significa, no entanto, que existe a distinta possibilidade de algumas destas conclusões serem inadequadas a um ambiente heterogêneo, uma vez que não são contempladas variáveis tais como diferentes tipos de autonomia, capacidade, funcionalidade e método operacional. Apesar de tudo, existem lições aprendidas nestes sistemas homogêneos, cuja utilidade transcende a referida barreira, uma vez que encontramos as mesmas noções perpetuadas nas equipas heterogêneas.

2.1.2 Heterogêneas

Contrariamente à sua homóloga homogênea, esta arquitectura de equipa apresenta mais pormenores, os quais devem ser tidos em atenção aquando do desenvolvimento do módulo de operações. Estes pormenores descendem directamente do facto de nos encontrarmos agora perante robôs com perfis diferentes (Figura 2.2) e com especializações particulares. Ao mesmo tempo que estas especializações permitem formar equipas mais abrangentes e poderosas, é previsível que acabem igualmente por levantar dificuldades na avaliação de performance do módulo, pois várias das métricas desenvolvidas para HRT homogêneas são difíceis de expandir para um ambiente heterogêneo [WL08], como por exemplo:

- Complicações de sobrecarga sensorial para os elementos humanos, devido à necessidade de se adaptar, durante a mesma missão, a modos diferentes de abordar os robôs a seu cargo, por causa da mudança de capacidades dos mesmo [MWKS06] [KWSN06] [YDS04];

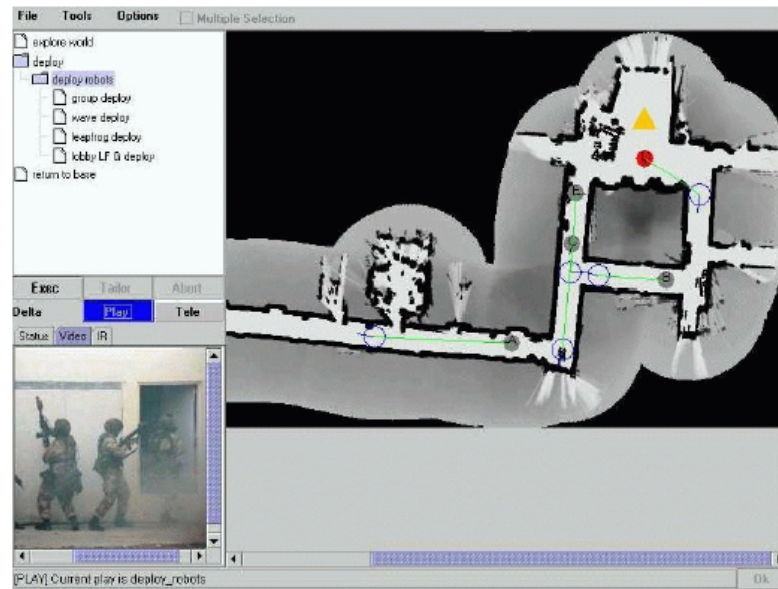


Figura 2.3: Exemplo da interface do sistema Macbeth [SAF⁺00]

- Complexidade na forma de representar e controlar a equipa, tomando em consideração que os diferentes níveis de autonomia e diferentes tipos de funções a desempenhar, pelos robôs, podem tornar necessário incluir componentes extra de software específicos para que tal seja possível [YDS04].

2.2 Sistemas Correntes

Tendo este tipo de equipas uma utilidade cada vez mais crescente no mundo real, não é inesperado detectar que existe já bastante trabalho efectuado na área de criação deste tipo de módulo de controlo. Seguem-se alguns sistemas já existentes.

2.2.1 Macbeth

O módulo Macbeth vem alargar o sistema do Alliance [SAF⁺00], cuja construção centrou-se em torno de uma arquitectura tolerante a falhas, onde a inoperabilidade de um elemento da equipa não compromete, por completo, a integridade do sucesso da missão [Par99]. No entanto, e contrariamente ao seu módulo inspirador, o Macbeth implementa uma interface que, apesar de não permitir o controlo directo da equipa em tempo real, deixa que o supervisor visualize um ambiente simples 2D no decorrer da operação. À semelhança do Alliance, todas as especificações de missão são carregadas previamente, não sendo permitido alterações a meio da execução desta.

Apesar da grande limitação, relativamente à impossibilidade de emissão directa de ordens à equipa robótica, é de notar que este módulo faz uso de uma interface simples,



Figura 2.4: Exemplo da interface do sistema MOCU [Pow06]

visível na Figura 2.3, que a torna um bom exemplo para análise, pois permite a supervisão e controlo de uma HRT média, através de um só módulo de operação.

2.2.2 MOCU

O MOCU (Multi-Robot Operator Control Unit) é um sistema desenvolvido para a Marinha dos Estados Unidos da América [Pow06] e, tal como pode ser observado na Figura 2.4, é um módulo que se apoia num conjunto alargado de monitores, de modo a possibilitar a supervisão de controlo de operações de uma HRT. Este sistema permite, contrariamente aos dois sistemas previamente referidos, o controlo em tempo real dos elementos robóticos a seu cargo.

No entanto, este controlo é obtido através do uso de hardware específico, que traz com ele complicações acrescidas em termos de:

- **Logística** - Num cenário de missão no exterior, seria caótica a montagem de um sistema desta envergadura. Isso desencorajaria o seu uso em missões mais complexas e de maior escala, o que derrotaria um dos objectivos desta dissertação - criar um módulo de operações que seja fácil de adaptar a novos e maiores cenários;
- **Usabilidade** - O grande número de controlos específicos e monitores de supervisão, apesar de possibilitarem uma panóplia de funcionalidades, tornariam complexa a formação de novos utilizadores, para além de poderem causar um excesso de "ruído".



Figura 2.5: Exemplo da interface de um jogo RTS

Dessa forma, apesar da abordagem obtida pelo MOCU ser considerada bastante interessante, o ideal seria obter o mesmo resultado, ou semelhante, sem a necessidade de hardware extra.

2.3 Paradigma de Jogos RTS

Como referido na introdução, o módulo de operações deve ser desenvolvido tendo como base conceitos oriundos de jogos RTS. Assim sendo, torna-se importante determinar qual a viabilidade do uso de jogos de estratégia como fonte de inspiração para a concepção de módulos de operação para HRT's.

Este tipo de abordagem possui os seus méritos, existindo já algumas aplicações em outras áreas a basearem-se nela, como [MWKS06] [JS01]. Para o projecto em mãos, o maior benefício de ter uma interface baseada em RTS é o aproveitamento de toda a experiência que a indústria de videojogos possui a criar interfaces intuitivas e de rápida aprendizagem. Para além disso, o estilo de jogo de RTS é bastante comum e conhecido mundialmente, sendo assim seguro assumir que esta abordagem será muito familiar para o controlador humano. Esta facilidade possibilita aumentar a SA do controlador humano, ao providenciar um ambiente já conhecido, habitado por estímulos familiares e uma forma de funcionamento sem grandes mistérios. Adicionalmente, este paradigma já foi utilizado,

com sucesso, na criação de módulos de controlo para *swarm robotics* [NGT⁺], assim como para melhorar a SA de um operador humano responsável por vários UV's [AT10]. Este último exemplo é-nos bastante útil, pois encontra-se dentro da área em que esta dissertação se baseia. Conclui-se assim que, determinar até que ponto é viável e exequível uma interface semelhante à da Figura 2.5, torna-se uma necessidade importante para poder coordenar os esforços de desenvolvimento. Assim sendo, é necessário abordar não só o modo de representação dos elementos da equipa de UAV's, como também a forma como será feita a interacção com essas representações.

2.3.1 Modos de Representação

Considerando que o objectivo pilar desta dissertação é a criação de um módulo de controlo que permita a fácil interacção com uma HRT de UAV's, heterogénea ou não, tentando no processo maximizar o número de robôs que um operador singular consegue gerir, sem que a sua performance seja posta em causa, rapidamente surge a necessidade de encontrar um local onde este tipo de provações sejam efectuadas, especialmente de forma comparativa com outras, em simultâneo e nas mesmas condições de operação, de forma a conseguir extrapolar quais as mais valias de cada abordagem e quais as suas fraquezas. Neste intuito, são organizadas provas robóticas de salvamento, provas estas que facultam uma fonte valiosa de informação referente a pormenores de HRI [MWKS06] [YDS04] [DSY03] e métricas de performance [Yan01]. Esses pormenores poderão ser cruzados com conceitos típicos de jogos RTS, de forma a encontrar pontos de partida sólidos para o desenvolvimento do módulo de operações.

2.3.1.1 Ponto de Vista

O primeiro pormenor de interacção que se revela importante é o ponto de vista ideal para o supervisor da HRT. Uma vez que nos encontramos cara a cara com uma realidade que pode ter como campo de operações uma temática extremamente diversa, quer em cenário (urbano, marítimo, deserto, etc), quer em tamanho (uma pequena rua de aldeia ou toda a extensão de um parque natural), torna-se necessário permitir ao controlador um ponto de vista vantajoso sobre este teatro de operações.

Assim sendo, será ideal que o ponto de vista da área de operações seja aéreo, permitindo ao operador uma visão alargada sobre todo o decorrer da missão. No entanto, pode ser necessário que o supervisor se aproxime dos veículos a seu cargo, alterando então o ponto de vista de uma visão isométrica a duas dimensões (Figura 2.6) para uma visão na primeira pessoa a três dimensões (Figura 2.7). Isto permite-nos chegar a um ponto importante, relativamente à representação do campo de operações.

Contrariamente à equipa de robôs presente em [MWKS06], os robôs deste projecto não possuem as mesmas capacidades de criação de mapas *on the fly* nem os sensores laser



Figura 2.6: Exemplo da interface desenvolvida em [HSS07], em modo de visão isométrica



Figura 2.7: Exemplo da interface desenvolvida em [HSS07], em modo de visão na primeira pessoa

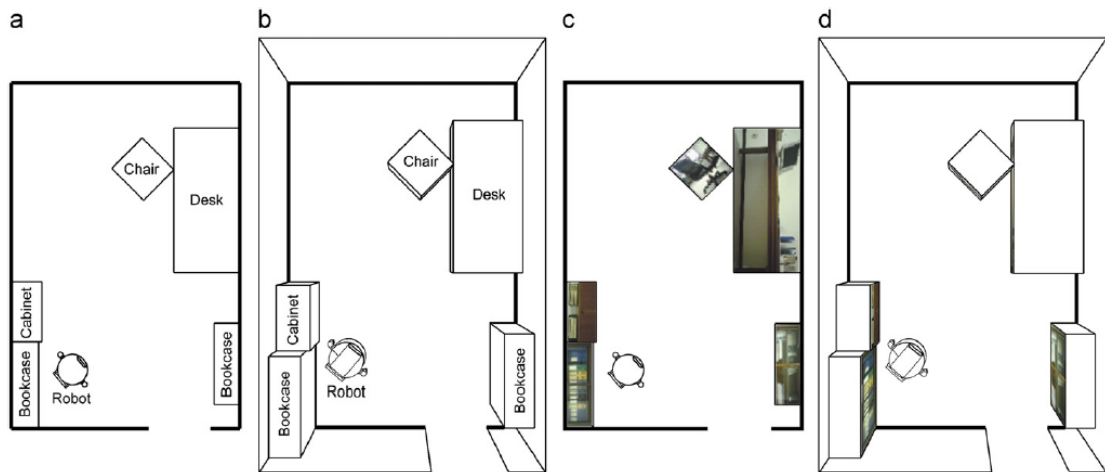


Figura 2.8: Exemplo dos vários tipos de mapas 2D: (a) 2D com rótulos; (b) 2D com elevações - 2,5D; (c) 2D com imagens reais; (d) 2D com imagens e elevações [LRY⁺07]

necessários para o efeito. Desta forma, o mundo terá de ser representado por um mapa, previamente obtido, da zona onde a missão irá decorrer. Isto significa que a passagem entre o 2D e o 3D tem de ser abordada para que seja possível determinar se será necessário incluir modelos tridimensionais nos mapas do módulo. A maior implicação relativamente a este ponto assenta no facto que este esforço de criação de mapas, com modelos tridimensionais correctos, irá aumentar a complexidade de preparação de uma missão, pois terão de ser criados mapas da área antes que a missão possa ser efectuada com segurança. Isto afectará a rapidez com que uma missão é aprovada e efectuada e, ao mesmo tempo, poderá causar a inutilização do modo 3D. Consideremos o seguinte cenário: caso o mapa seja demasiado complexo para ser totalmente modelado, dentro do prazo de preparação para a missão, isso irá tornar o modo 3D pouco seguro, devido à ausência de pormenor, o que remeterá o supervisor ao modo 2D. Tendo em conta que a localização da área de missão pode não ser conhecida com tempo suficiente para a criação de um modelo correcto, foi determinado que seria adoptada uma abordagem 2D relativamente à construção do mapa, visto esta ser mais simples de conceber, dados os recursos disponíveis.

2.3.1.2 Mapa

Dentro da representação a duas dimensões, surge ainda a questão relativamente ao "realismo" que irá ser concedido ao mapa. Da mesma forma que uma área pode ser representada através de imagens detalhadas dos elementos que a constituem, é também possível representá-la utilizando modos simplificados, como os de um mapa cartográfico. Estes pormenores são abordados em [LRY⁺07], onde são apresentadas várias formas de representar ambientes em duas dimensões, como é possível verificar na Figura 2.8.

Assim sendo, existem várias formas de expandir este 2D primitivo, como por exemplo:

- Através da adição de rótulos identificativos dos elementos representados no mapa;
- Com a substituição dos contornos dos elementos por imagens reais dos mesmos, tentando assim conferir um reconhecimento mais imediato;
- A partir da simulação de relevo em alguns elementos do mapa, passando assim para um paradigma 2,5D.

O trabalho desenvolvido em [LRY⁺07] demonstrou-se instrumental para melhor compreender a importância destes factores, bem como quais as opções mais apelativas a seguir. Dito isto, será seguido então um modelo 2D, preferencialmente apoiado em imagens reais dos elementos presentes no mapa. Esta decisão é igualmente influenciada pela configuração actual do Neptus que permite a representação de mapas através de imagens realistas, obtidas através do Google Earth.

2.3.2 Modo de Interação

Determinados os métodos de representação, é necessário agora alinhar o método através do qual irá ser feita a interação com o módulo de operações. Tipicamente, esta interação pode ser efectuada através do normal controlo de rato, no entanto, é possível optar por um método, igualmente fiável, de toque. Após um período de reflexão, foi determinado que ambos os métodos não necessitam de se excluir mutuamente. Desta forma, será possível permitir que ambas as formas de interação coexistam, conseguindo assim que haja, inclusive, alternativas de controlo, caso um dos métodos falhe devido a problemas técnicos. Relativamente ao método táctil de controlo surge, obviamente, a preocupação referente à precisão entre a posição tocada no ecrã e as coordenadas verdadeiras no campo de operações. Segundo o estudo efectuado sobre dispositivos portáteis com ecrãs de pequenas dimensões [LRY⁺07], o erro em pouco aparenta afectar a noção que o utilizador tem do local que está a indicar e o verdadeiro local no mundo real. Especula-se então que, num ecrã maior, apesar do aumento do campo de operações, o erro dever-se-á manter dentro de um intervalo aceitável. Admitindo que ambos os métodos são fiáveis, resta somente correlacionar este facto com a arquitectura actual do Neptus, de forma a determinar se existe algum impeditivo para a implementação de ambos estes esquemas de controlo. Visto existirem certas restrições ao desenvolvimento no Neptus (restrições estas que serão aprofundadas no Capítulo 3), prevê-se que o desenvolvimento de implementação de um método de controlo táctil irá condicionar bastante o tempo de desenvolvimento do módulo de operações. Conclui-se então que, para impedir desvios que possam por em causa o sucesso dos objectivos principais desta dissertação, o método de interação utilizado pelo supervisor dentro o módulo de operações será a do normal rato e teclado.

$$FO = \frac{AT}{IT}$$

Figura 2.9: Fórmula simplificada da métrica *fan-out* [OW04]

2.4 Métricas de Avaliação

Visto termos centrado o nosso trabalho na interface do módulo de operações, graças à abstracção trazida pelo Neptus a nível de arquitectura e comunicação com os UAV's, isto significa que teremos apenas de ter em conta como avaliar a forma como a nova interface afecta a performance do supervisor no decorrer da sua tarefa. Assim sendo, torna-se necessário observar uma série de noções e métricas fundamentais para esta avaliação, visto ainda não existe uma métrica central para a determinação absoluta da performance de um HRT [Yan01].

2.4.1 Fan-Out

Apesar de ser bastante complexo obter uma métrica única comum que avalie uma HRT na íntegra, devido à sua natureza multi-modal, é, no entanto, possível determinar um conjunto de métricas que analisem a eficácia de várias das partes constituintes da HRT [CC07], permitindo dessa forma obter uma ideia da performance do todo. O *fan-out* não é mais do que o número máximo de robôs que um supervisor consegue gerir antes que a performance da equipa comece a descer [OW04] devido à falta de capacidade do supervisor dar resposta aos pedidos que surgem. Esta será uma das métricas que nos permitirá avaliar o novo módulo de operações, desenvolvido sobre o Neptus.

O *fan-out* apoia-se na medição empírica de outras variáveis, as quais constituem a sua fórmula de cálculo (Figura 2.9). Devido às diferenças entre equipas homogéneas e heterogéneas, é bastante complexo determinar uma fórmula definitiva de cálculo para o *fan-out*. No entanto, existem elementos da fórmula que são comuns nas suas várias encarnações:

- **NT** (*neglect time*), que não é mais do que o tempo que o robô pode ser ignorado em segurança antes que a sua performance decresça;
- **AT** (*activity time*), que consiste no tempo efectivamente activo após receber comandos por parte do controlador;
- **IT** (*interaction time*), que determina o tempo que o utilizador demora a interagir com o robô.

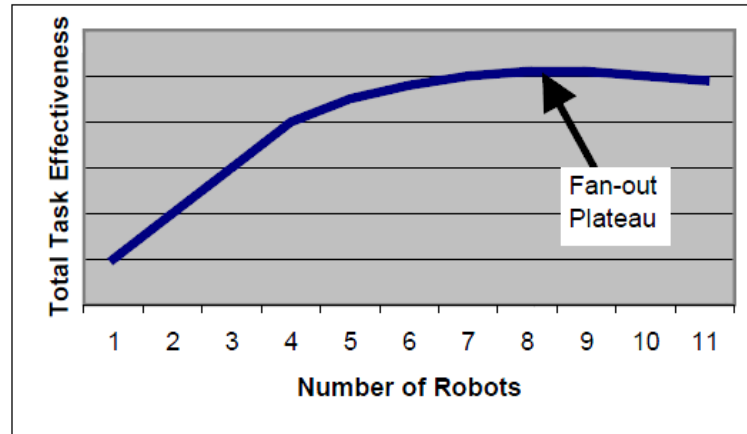


Figura 2.10: Limite atingido de *fan-out* [OW04]

Determinando o FO, é então possível observar se houve melhoria ou não entre cada interacção do processo de desenvolvimento. Tipicamente, caso seja tomado um passo positivo no desenvolvimento da interface, isso será reflectido num aumento do FO, se for registado um aumento do AT ou uma diminuição do IT do utilizador. Obviamente, a autonomia dos robôs ditará igualmente um aumento ou decréscimo do FO; no entanto, todos os UAV's presentes no LSTS possuem o mesmo grau de autonomia, garantindo assim consistência nesta componente da avaliação. Apesar disso, este aumento de FO atinge, eventualmente, um máximo a partir do qual estabiliza, visível na Figura 2.10.

Outra abordagem ao FO é apresentada em [WL08], sendo este um modelo que aprofunda o conceito da NT que, tal como o nome evidencia, significa o tempo de tolerância que pode ser dada a um robô que esteja a ser ignorado pelo operador, antes que a sua performance comece a baixar.

Este método recorre a mais detalhe, dissecando o NE, visível na Figura 2.12, em componentes mais pequenas como:

- **OT** (*occupied time*) - que representa o tempo gasto a efectuar a tarefa pedida;
- **FT** (*free time*), que consiste no tempo livre antes do início da tarefa e logo após a sua conclusão.

$$CD = 1 - \frac{\sum FT}{NT} = \frac{\sum OT}{NT}$$

Figura 2.11: Variante do calculo para o *fan-out* [WL08]

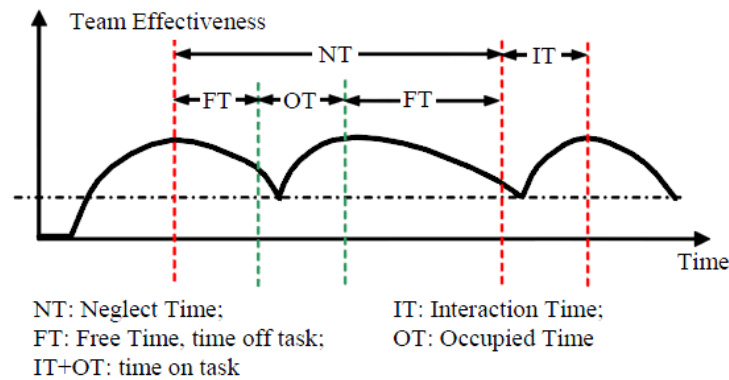


Figura 2.12: Extensão do conceito de *neglect time* [WL08]

2.4.2 Situation Awareness

A *situation awareness* não é mais do que o grau de atenção que o controlador consegue ter do ambiente a ser supervisionado e do estado de cada robô da equipa [SFK⁺06]. Dito isto, é fácil assumir que a qualidade da fusão de informação e a simplicidade da interface irão transparecer um grau superior de *situation awareness*. Assim sendo, torna-se óbvio que será necessário prestar grande cuidado no desenvolvimento da interface gráfica, apoiando o desenvolvimento em estudos já realizados [Ada02] [Nie94]. Da mesma forma, irão ser seguidos os concelhos presentes em [KWSN06] para a utilização de outras formas de estímulo sensorial como método de transmitir informação ao controlador. É necessário recordar que é possível transmitir uma grande gama de informação em simultâneo, recorrendo a métodos como o som ou cor, complementando, ou até mesmo substituindo, as típicas representações numéricas.

Um bom exemplo deste facto está presente em [KWSN06], onde, ao diversificar a forma como a informação é apresentada, esta demonstrou um nível superior de assim-

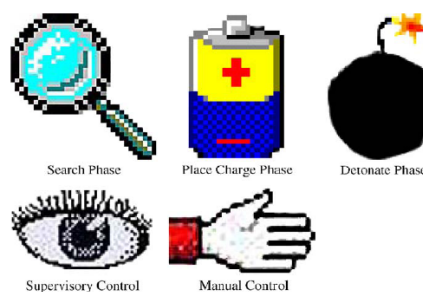


Figura 2.13: Exemplo de ícons utilizados em [KWSN06] para a investigação dos benefícios da abordagem multi-modal

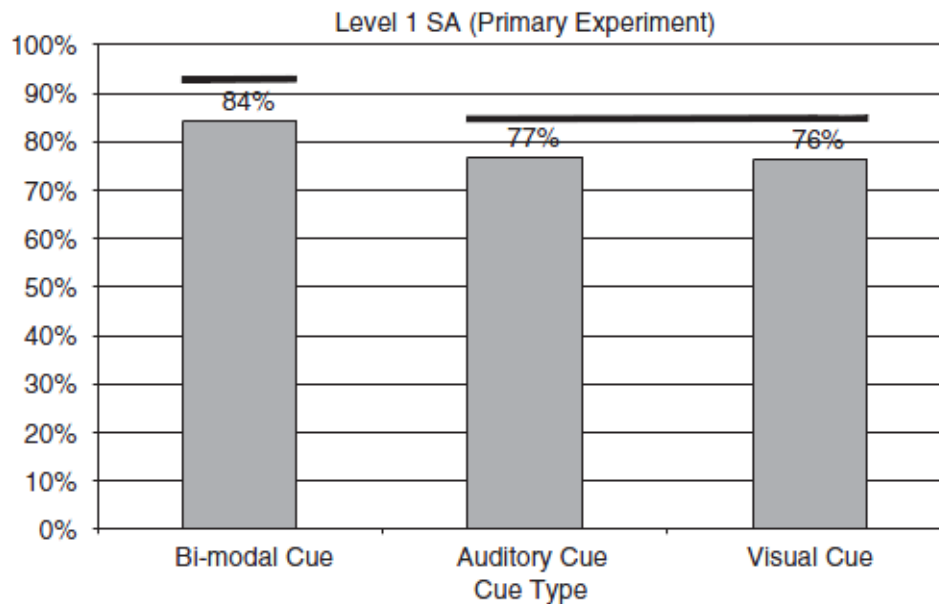


Figura 2.14: Resultados obtidos referentes aos benefícios da abordagem multi-modal [KWSN06]

ilação, devido ao elevado grau de facilidade com que o operador a conseguia adquirir (Figura 2.14); ou seja, ao recorrer a outros métodos que não apenas o texto corrido, é possível obter um aumento da SA que, por sua vez, torna a aplicação mais robusta e apta a lidar com crescentes adições ao número total de UAV's da HRT.

Com isto, tentar-se-á obter um elevado grau de *situation awareness*, visto este influenciar directamente o nível de performance que é desejado do módulo de operações. No entanto, é necessário que exista a possibilidade de medir esta variável, de forma a comprovar se a implementação feita no módulo de operações está de facto a ser benéfica para o aumento do SA ou se, pelo contrário, nenhuma alteração se verifica neste.

2.4.2.1 Método SAGAT

Para medir a SA, encontra-se uma panóplia de métodos distintos; no entanto, para cenários com veículos aéreos supervisionados por um só ser humano, é recorrente o uso de dois. Um deles, denominado SAGAT (Situation Awareness Global Assessment Technique) [KPS⁺06], é um método objectivo de avaliação, contrariamente ao outro, denominado SART (Situation Awareness Rating Technique) [BB09], que é um método subjectivo de avaliação. Por uma questão de precisão de dados [SSW⁺09], foi optado o uso do método SAGAT como forma de determinar a SA do supervisor durante o uso do módulo de operações. O método SAGAT tem já um largo uso na avaliação da SA em vários ambientes, não só o da robótica [ATP95]. No entanto, a sua aplicação é bastante comum na avaliação da SA em cenário de controlo aéreo.

2.4.3 Workload

O *workload* é uma métrica recorrente na avaliação da performance de HRT's [PJS⁺10] [Miy01] [CM07] e representa a carga sentida - significando que é uma métrica subjectiva - pelo supervisor no decorrer normal de uma missão. Sendo uma métrica subjectiva, implica que diversas variáveis condicionam a sua medição, sendo assim necessário tomá-las em consideração para podermos ter uma visão unificada do *workload* sentido pelo elemento humano da equipa HRT. De forma a solucionar este factor, será utilizado o método desenvolvido pela NASA (National Aeronautics and Space Administration) - NASA-TLX (NASA Task Load Index) -, que consegue repartir o *workload* em seis componentes distintas [WGG10].

2.4.3.1 Método NASA-TLX

Desenvolvido pela NASA, o método NASA-TLX é tipicamente utilizado para medir o *workload* sentido pelo sujeito avaliado durante tarefas consideradas de grande responsabilidade e complexidade [MDKY09] [SFK⁺06] [DN08]. Este *workload* total é calculado após a divisão do mesmo em seis componentes específicas: Exigência Mental; Exigência Física; Exigência Temporal; Performance; Esforço; Frustração. Efectuada esta divisão, é necessário estabelecer o peso de cada componente para ser depois possível determinar qual a componente mais significativa no cálculo da *workload* total. Para uma visão mais detalhada do processo NASA-TLX, pode ser consultado o Anexo B.

2.5 Resumo

Neste capítulo, foi feita uma introdução aos diferentes tipos de HRT's existentes, bem como a apresentação de sistemas actualmente utilizados para controlo e supervisão das mesmas. De seguida, foi mencionado o paradigma de jogos RTS e o contributo que irá ter para o projecto, principalmente a forma como será abordado. Finalmente, é feita uma exibição das métricas que serão utilizadas para avaliar a performance do módulo de operações a ser implementado.

De seguida, é feita a apresentação do Neptus e da sua arquitectura como ponto de partida para o projecto.

Capítulo 3

Análise da Solução em Neptus

Antes de se avançar com a implementação do módulo, devemos fazer uma análise cuidada ao cenário que nos é apresentado. Como tal, é necessário fazer um apanhado de todas as variáveis que afectaram o processo e concepção do módulo de operações.

3.1 Neptus

Uma vez exposto o estado da arte que envolve este tema, torna-se agora necessário abordar, pormenorizadamente, a ferramenta que será utilizada como a base para o desenvolvimento da consola de supervisão para UAV's - o Neptus.

3.1.1 Introdução

O Neptus, visível na Figura 3.1, é uma criação do Laboratório de Sistemas e Tecnologia Subaquática (LSTS) e tem vindo a ser adaptado, há já algum tempo, para uma maior compatibilidade com as necessidades da equipa de UAV's presente no LSTS, os AsasF. A ferramenta é concebida em Java e está em constante expansão, através dos esforços de uma equipa de desenvolvimento com o mesmo nome que a *framework*. O Neptus permite a interacção com vários tipos de veículos, quer sejam autónomos, semiautónomos ou teleguiados e, apesar do seu foco ser em controlo e supervisão marítimo, o Neptus apresenta também uma base sólida para o desenvolvimento, com sucesso, de uma consola para supervisão de UAV's, sendo possível o reaproveitamento de vários componentes já existentes para veículos marítimos. Actualmente, o Neptus é já utilizado em missões dos AsasF, mas a sua implementação, relativamente às necessidades do supervisor, são meramente soluções ad hoc. Seguidamente, é apresentado um breve esquema da arquitectura do Neptus, relevante para o processo de criação de um novo módulo visual.

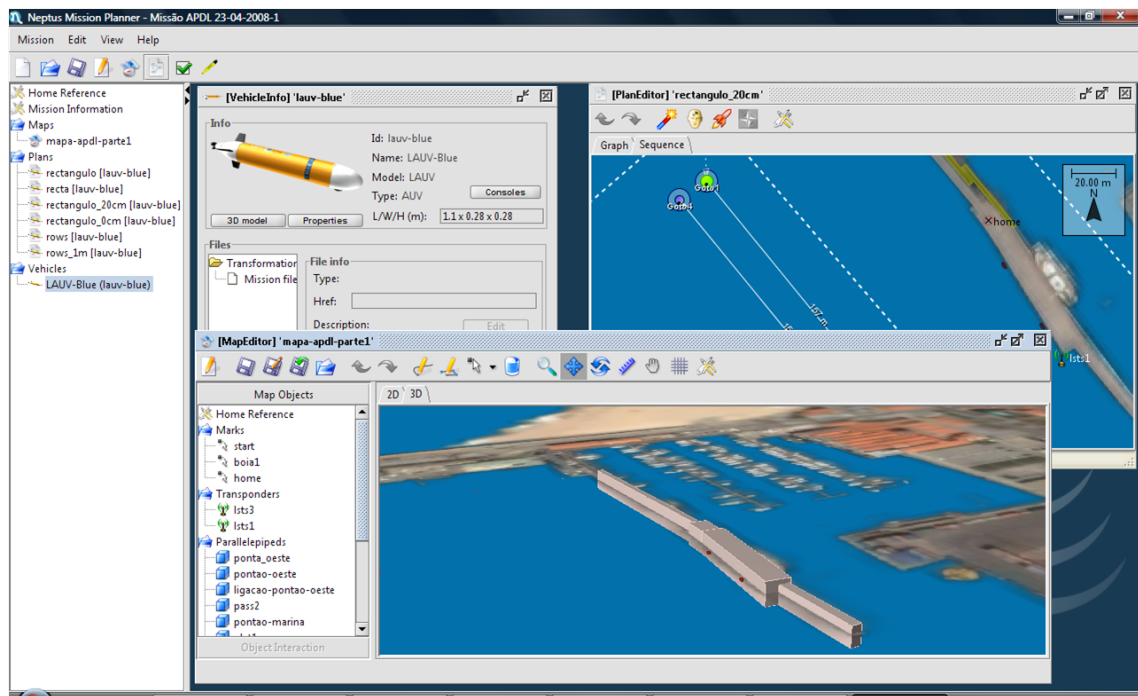


Figura 3.1: Exemplo da interface do módulo Neptus

3.1.2 Arquitectura

Para poder facilitar a sua manutenção e expansibilidade, o Neptus é concebido de forma modular, tentando manter ao máximo a independência entre cada novo módulo criado. Como é possível verificar na Figura 3.2, o Neptus, na sua essência, é composto por um módulo base onde residem os métodos mínimos necessários para a implementação de uma consola e vários módulos externos, denominados *plugins*, que podem ser adicionados a uma consola base, fornecendo assim novas capacidades à mesma. Uma vez que o Neptus se trata de uma ferramenta em constante expansão e reestruturação, esta forma arquitectural permite reduzir a um mínimo qualquer tipo de repercussões que novo código, em forma de *plugin*, possa causar a módulos já implementados e testados para utilização em missões.

No entanto, o mesmo não pode ser dito de alterações feitas a fragmentos do módulo central do Neptus pois, como é possível verificar na Figura 3.3, qualquer alteração a este módulo poderá afectar de forma nefasta consolas já estabelecidas e consideradas estáveis. A relevância deste ponto para o projecto em curso assenta no pedido expresso da não alteração de código existente nesta zona da arquitectura. Isto leva, por um lado, a uma conservação da integridade e segurança da versão do Neptus em curso, enquanto que, por outro lado, causa um aumento na complexidade do desenvolvimento de novo software complementar à ferramenta. De igual forma, esta condição pode levar à criação de métodos cujas funcionalidades se encontram num estado abaixo do que seria consider-

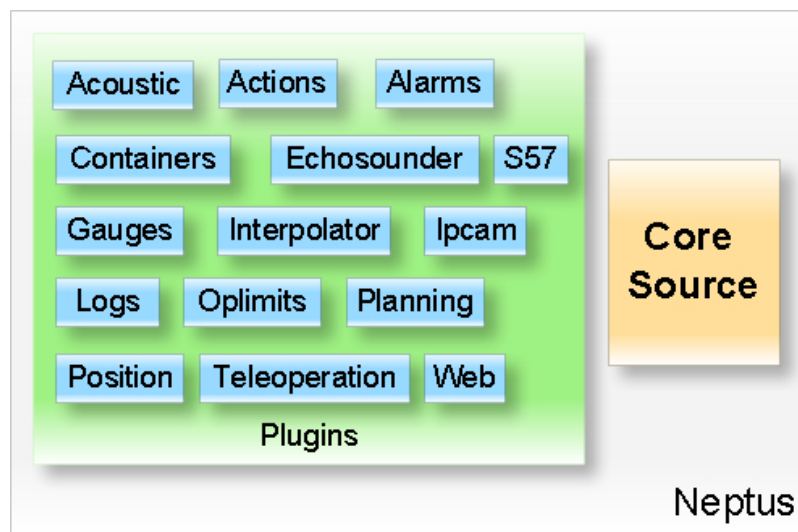


Figura 3.2: Arquitectura simplificada do código-fonte do Neptus

ado óptimo, devido à impossibilidade de alterar directamente zonas de código, pois estas encontram-se alojadas no módulo central do Neptus.

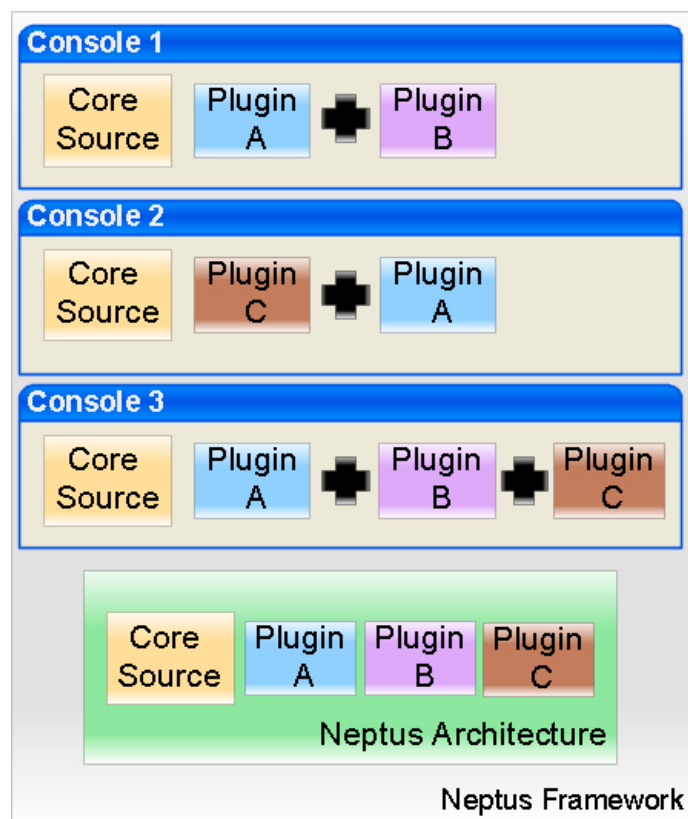


Figura 3.3: Exemplo da constituição das consolas Neptus

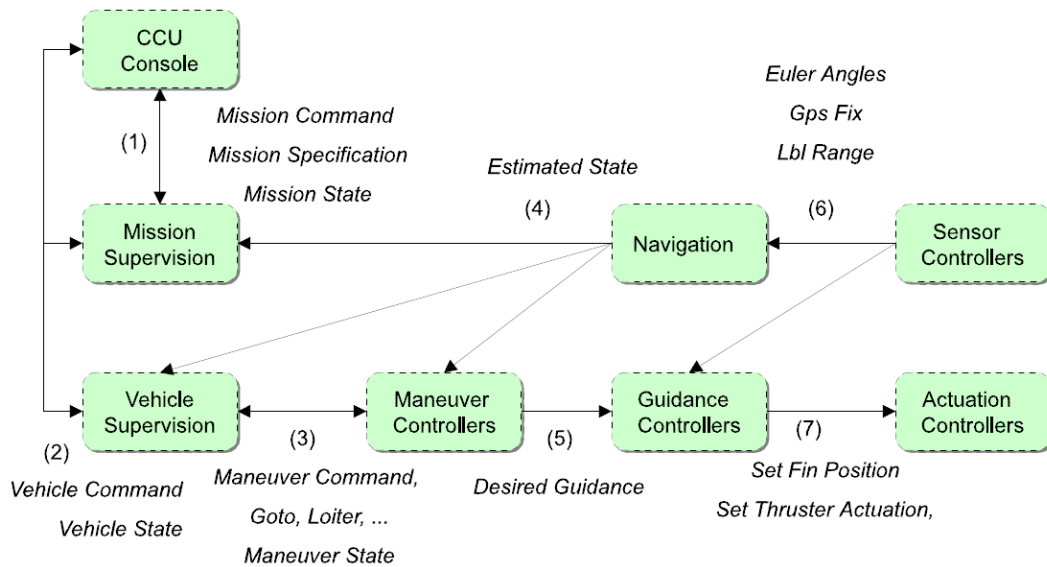


Figura 3.4: Diagrama exemplo de mensagens IMC [MDM⁺09]

3.2 IMC

O IMC (Inter-Module Communication Protocol), foi desenhado e implementado pelo LSTS e é um protocolo de comunicação preparado para veículos autónomos [MDM⁺09], podendo estes ser terrestres, marítimos ou aéreos. O protocolo foi concebido como um *middleware* entre sensores, operadores e veículos activos de forma a fornecer uma camada transparente entre estes. O IMC é compatível com standards internacionais, sendo exemplo disso o STANAG 4586 [Org07].

O IMC tem o seguinte conjunto de tipos de mensagem:

- Mensagens de controlo de missões;
- Mensagens de controlo de veículos;
- Mensagens de manobras;
- Mensagens de orientação;
- Mensagens de navegação;
- Mensagens de sensores;
- Mensagens de actuadores.

Desta maneira, é possível um veículo comunicar com vários tipos de estações terrestres e até mesmo ser operado por diferentes pessoas no espaço de tempo. Existe

ainda a capacidade de se integrar completamente com o Neptus, o que faz com que nesta plataforma seja possível coordenar vários veículos em simultâneo.

3.3 Dune

O Dune é um sistema desenvolvido no LSTS e é o software que vai a bordo do UAV [dun07]. Este software corre num pequeno computador, denominado PC-104 [pc1], que tem uma capacidade de processamento mais reduzida, devido ao seu diminuto tamanho. Esta ferramenta é responsável por comandar o UAV e contém *drivers* para sistemas de aquisição de informação, navegação, controlos de manobras e para simulação com *hardware-in-the-loop*. Este sistema corre numa *compact flash* que vai ligada ao PC-104 e, uma vez o computador ligado, este programa arranca automaticamente.

Logo que se encontra em voo, o Dune está responsável por comandar o Piccolo e gerir a comunicação com terra. Adicionalmente existe a possibilidade de ser ligado ao Dune uma vasta gama de dispositivos essenciais a diferentes tipos de missões. Como exemplos desses dispositivos temos:

- Dispositivos de vigilância vídeo;
- Radares;
- Sensores atmosféricos;
- Etc.

O Dune comunica com a equipa no solo através do protocolo IMC, e utiliza o standard de mensagens da CloudCap para comunicar com o Piccolo.

3.4 Piccolo

Piccolo é um sistema de controlo, completo e integrado, para pequenos UAV's, vendido e produzido pela empresa CloudCap [pic] e é composto por vários componentes de hardware e software. Este sistema contém a capacidade de cálculo de todas as variáveis físicas de um UAV e também de cruzá-las com as informações adquiridas dos sensores internos. O Piccolo possui, entre vários componentes, um sistema GPS e um sistema de rádio para comunicação.

O Piccolo, visível na Figura 3.5, na sua essência não é mais do que o sistema autónomo de pilotagem UAV; a ele estão ligados todos os actuadores do UAV e, através do modelo 3D do avião, este consegue pilotar o UAV de forma segura e estável. É igualmente possível ligar, via rádio, o Piccolo a uma *ground station* (GS), que corre um programa próprio



Figura 3.5: Imagem de um exemplar Piccolo [pic]

de controlo desenvolvido pela empresa que comercializa o sistema. Este programa possibilita o comando e acompanhamento da missão, tal como o Neptus. Possibilita ainda a criação de simulações de voo, injectando os valores físicos directamente no Piccolo, fazendo-o pensar que está em voo real.

3.5 Ambiente entre o Software de Missão

Como já foi referido anteriormente, o Neptus, tal como o nome deixa transparecer, teve um desenvolvimento inicial mais focado em veículos marítimos, o que levou a que este não fosse, à partida, perfeitamente compatível com as necessidades dos operadores aéreos dos AsasF. Assim sendo, foi crucial encontrar uma forma de colmatar estas necessidades, através da utilização de outro software que, juntamente com o Neptus, permitisse a execução de missões com UAV's por parte do LSTS. É neste contexto que surge o Piccolo Command Center (PCC) [pcc], como ferramenta de controlo dos UAV's.

Tipicamente, para a supervisão e controlo de veículos marítimos, são necessários apenas três grandes componentes: Neptus; IMC; e Dune. Com estes três elementos, é possível dar resposta às necessidades do LSTS em missões marítimas. No entanto, no caso de missões UAV, é necessário adicionar um novo componente a este grupo, o PCC. De forma a ser possível executar missões com UAV's, é necessário adicionar o PCC como ferramenta para operar o UAV remotamente, relegando o Neptus para um papel de supervisão. Naturalmente, este cenário deseja-se temporário, contribuindo para isso projectos como este, que expandem o Neptus de forma a suportar funcionalidades semelhantes à do PCC. Idealmente, o Neptus substituirá na íntegra o PCC, fazendo com que as missões de UAV's se assemelhem às de veículos marítimos relativamente ao software utilizado.

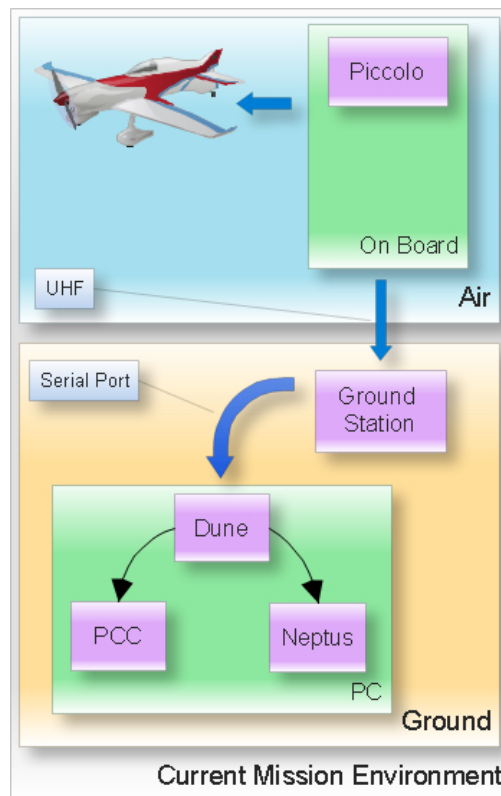


Figura 3.6: Esquema a actual interacção entre o software necessário para o voo de um UAV

Actualmente uma missão de UAV's, a nível de software, divide-se entre várias aplicações e protocolos de comunicação. Como é visível na Figura 3.6 no UAV apenas se encontra o Piccolo, para efeitos de controlo e navegação, sendo este responsável pelo contacto com a GS. A GS por sua vez transmite os dados recebidos para uma máquina com Dune que, após os receber, faz a disseminação dessa informação para o Neptus, através de IMC, e para o PCC, através do standard de mensagens da CloudCap.

O objectivo final é atingir um modelo igual ao representado na Figura 3.7 onde dentro de cada UAV irá constar, para além do Piccolo, um computador PC-104 ligado a um módulo Dune. Este Dune móvel fará então a comunicação com a GS onde estará um módulo Neptus pronto a receber e enviar informação, dessa forma eliminando o software adicional PCC.

3.6 Metodologia de Operações com UAV's

De forma a melhor compreender as necessidades do projecto, torna-se necessário não só tomar em consideração a forma como o software actual é constituído, mas também como interage entre si. Torna-se igualmente fundamental observar como decorre uma

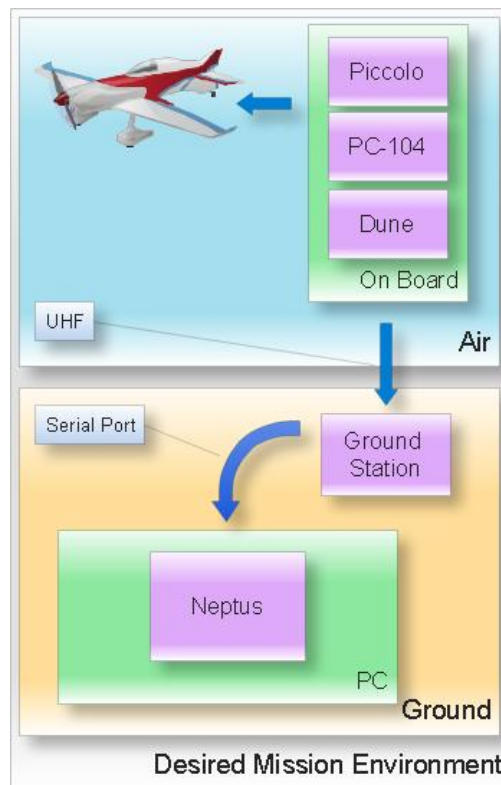


Figura 3.7: Esquema da desejada interação entre o software necessário para o voo de um UAV

típica missão com UAV's, de forma a melhor aplicar, durante o processo de criação da consola, os princípios do paradigma RTS adoptado. A forma como as missões são conduzidas introduzem variáveis que não podem ser ignoradas nem descartadas, caso contrário incorremos no risco de desenvolver um módulo de operação inviável à equipa que o irá utilizar.

Numa missão existem três elementos principais:

- **Supervisor** - Responsável pela supervisão da equipa na sua totalidade. Este elemento é o único que sabe, em qualquer instante, o estado de todos os UAV's no ar e recai sobre ele a responsabilidade de coordenação e segurança da missão como um todo;
- **Operador** - Responsável por supervisionar o UAV a si indicado, dando-lhe novas tarefas e supervisionando as que já estão a decorrer. É também função deste elemento monitorizar a integridade física do UAV durante o decorrer de uma missão;
- **Piloto** - Responsável pelo controlo manual, caso seja necessário, do veículo aéreo.

Tal como pode ser observado na Figura 3.8, o supervisor da equipa, actualmente, não tem qualquer poder directo na sua consola para alterar planos de missão dos UAV's à sua responsabilidade. Essa capacidade encontra-se nas consolas PCC, cada uma agregada

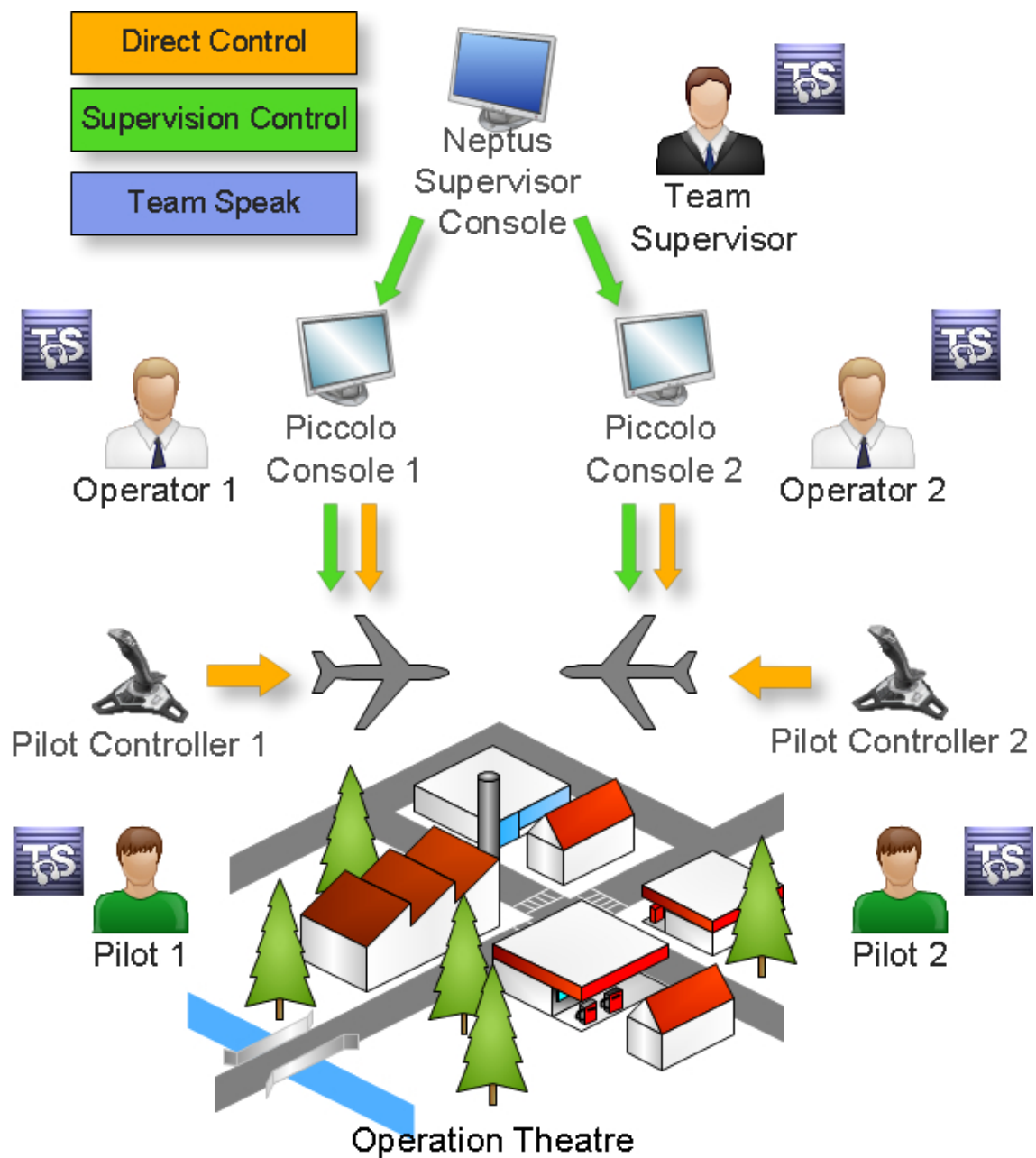


Figura 3.8: Esquematização dos elementos participantes numa missão de UAV

ao seu veículo, e nos pilotos. No entanto, é necessário existir uma forma de permitir ao supervisor efectuar as correcções e alterações necessárias ao plano da missão, quer devido a razões de segurança, quer devido a reformulação de objectivos. Para esse efeito, é usado software externo VoIP, o TeamSpeak.

Utilizando o TeamSpeak, todos os elementos da missão conseguem acatar as decisões do supervisor da equipa, bem como compreender as necessidades de outros operadores em campo. Este tipo de coordenação não é possível de outra forma, pois cada UAV

executa a sua missão sem saber se existem outros UAV's no ar, daí o elevado grau de responsabilidade e stress do supervisor de equipa. O módulo de operação criado tem de abordar estes pormenores de interacção entre elementos de equipa de forma a não aumentar a entropia entre eles.

3.7 Requisitos do Problema

De forma a orientar o processo de desenvolvimento, foram recolhidos requisitos necessários ao sucesso do projecto. Com esses requisitos recolhidos, é possível determinar marcos a atingir, bem como a sua prioridade na conclusão.

3.7.1 User Story

O objectivo do sistema é monitorizar uma área de grandes dimensões usando, para isso, uma equipa de UAV's e vários sensores no solo. Todos os UAV's têm uma câmara fotográfica a bordo que lhes permite capturar imagens de alta resolução sobre uma área. Dependendo da altitude do UAV, a resolução das imagens pode ser maior ou menor (bem como a área de cobertura).

No solo, alguns sensores estão espalhados e são capazes de detectar movimento. Sempre que um movimento é detectado é emitido um alerta para a rede, juntamente com a posição geográfica do sensor. Os UAVs devem ser então utilizados para fotografar as zonas onde sejam emitidos alertas.

Na estação base existem vários operadores humanos:

- Um ou mais operadores têm a tarefa de associar UAV's a inspecções, escolhendo a altitude de inspecção;
- Um ou mais pilotos são usados para descolar e aterrar UAV's usando controlo manual;
- O supervisor cataloga as imagens recolhidas pelos UAV's como "Falso alarme", "Ameaça" ou "Imperceptível";
- O supervisor de missão deve igualmente garantir que todos os sistemas funcionam de forma segura (combustível suficiente, alcance de comunicações, ...);
- O supervisor pode ainda escolher ignorar alguns dos alertas no solo;
- O supervisor pode tomar o controlo (voo autónomo) de um ou mais UAVs, podendo comandar a sua aterragem ou envio para outros *waypoints*;
- O supervisor pode também comandar a descolagem de novos veículos.

3.7.2 Requisitos Funcionais

Da *user story* anterior, é possível extrair casos de uso necessários ao projecto em questão.

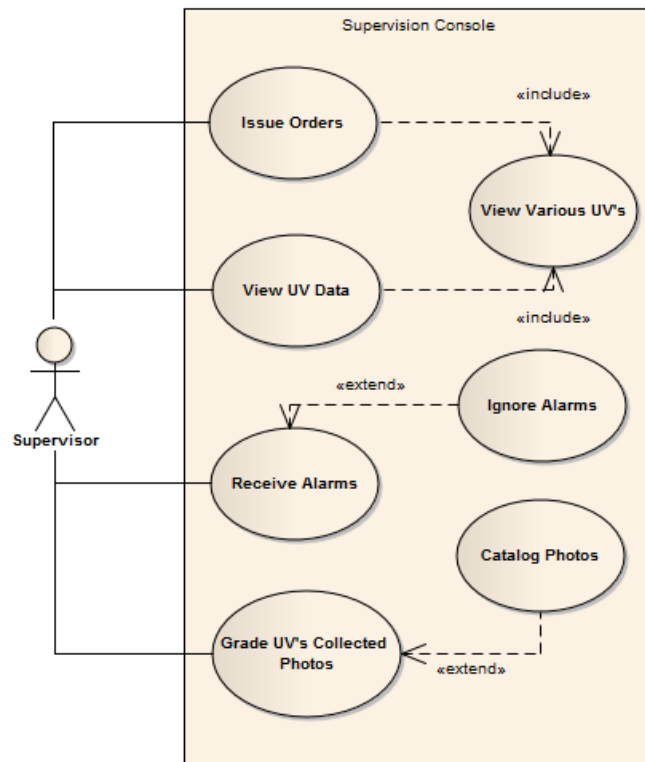


Figura 3.9: Diagrama de Casos de Uso do módulo desenvolvido

Observando a Figura 3.9, é fácil detectar que existe agora um conjunto de situações que necessitam de ser abordadas pelo módulo de controlo. Correlacionando estes casos de uso com a análise efectuada ao Neptus, ao modo de operação de uma missão de UAV, e à pesquisa efectuada, é então possível determinar um método para abordar cada um deles.

- **Emitir Ordens** - A emissão de ordens através da consola torna-se complexa devido à arquitectura do Neptus, que coloca o supervisor como um receptor numa situação de *broadcast* de várias outras consolas em operação. Aliando a isso o facto de todo o módulo de comunicação se encontrar dentro da secção base do código Neptus, e de esta estar sob restrições quanto à alteração dos seus métodos, é extremamente difícil alcançar este objectivo dentro da consola. No entanto, considerando que no decorrer de uma missão todos os elementos participantes estão em constante contacto através da ferramenta VoIP, é possível remeter este caso de utilização para a ferramenta TeamSpeak.

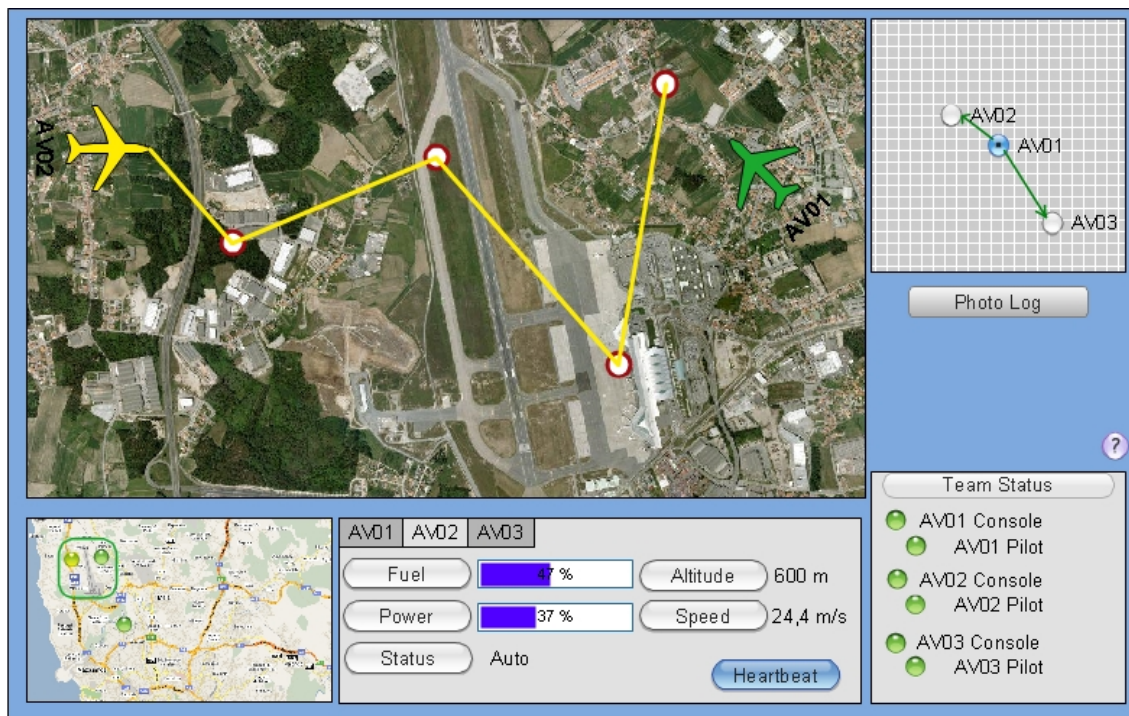


Figura 3.10: Exemplo da interface principal da consola a desenvolver

- Visualizar vários UAV's** - A visualização de vários UAV's numa só consola é já possível dentro do *PlanningPanel*. No entanto, a forma como eles são representados (utilizando uma imagem realista do veículo) limita a informação que, só de observar o ícon do veículo, poderia ser retirada. Como alternativa, este ícon realista será substituído por um outro, desta vez dinâmico, que irá permitir ao supervisor aperceber-se do grau de atenção que o veículo necessita, sem que este tenha de percrutar vários *displays* numéricos, como se pode observar na Figura 3.10. Desta maneira, conseguimos fundir informação utilizando os ensinamentos dos benefícios multi-modais de [KWSN06].
- Visualizar dados de UAV** - De forma a visualizar os dados referentes ao UAV seleccionado, irão usar-se *Generic Displays* já utilizados dentro da plataforma Neptus. Serão, no entanto, adicionados painéis novos que permitirão a visualização e o *cross-referencing* de outros dados que até agora não era permitido pelo Neptus. Como exemplo, temos o estado individual de cada componente do UAV (*flaps*, leme, etc.) ou até mesmo a visualização simultânea de todos os níveis de altitude de cada UAV no ar.
- Receber Alarmes** - Para a recepção de alarmes, vindos de sensores no solo, será usada a mesma abordagem utilizada para a visualização dos UAV's.



Figura 3.11: Exemplo do funcionamento da galeria de fotos recolhidas pelos UAV's

- **Avaliar Images Recebidas de UAV** - Para permitir ao supervisor avaliar as imagens recolhidas pelos UAV's em voo, será adicionado um novo painel, incorporado na consola, que permitirá percorrer as fotos e classificá-las *on the fly*. Este painel será activado somente se o supervisor desejar ver as fotos recolhidas, caso contrário ele permanecerá escondido, desocupando assim o campo de visão do supervisor durante a missão. Isto é visível na Figura 3.11.

Efectuado o levantamento dos requisitos operacionais do projecto, torna-se então necessário fixar os seus níveis de prioridade, para melhor orientar o ênfase de desenvolvimento. Isto observa-se na Tabela 3.1.

Tabela 3.1: Lista de prioridades dos requisitos funcionais

Requisito	Prioridade
Emitir Ordens	Baixo
Visualizar vários UAV's	Alta
Visualizar dados de UAV	Alta
Receber Alarmes	Média
Avaliar Images Recebidas de UAV	Média

Tabela 3.2: Lista de prioridades dos requisitos não funcionais

Requisito	Prioridade
Reduzir <i>workload</i> do supervisor	Alto
Aumentar a SA do supervisor	Alta
Aumentar o FO de uma missão	Média
Manter a curva de aprendizagem para o módulo reduzida	Baixa

3.7.3 Requisitos Não Funcionais

Para além dos requisitos relativos às funcionalidades que devem ser implementadas no módulo de operações, é necessário também abordar aqueles que são externos ao desenvolvimento do código e que se centram na forma como o módulo de operações se deve comportar após a implementação das funcionalidades previstas.

De forma a cumprir os requisitos da Tabela 3.2, terão de ser observadas as lições aprendidas no Captítulo 2. Para alcançar o aumento do SA e do FO, deve ser maximizada a fusão de informação apresentada ao supervisor. Adicionalmente, uma excelente forma de manter a curva de aprendizagem reduzida é basear a abordagem de construção do módulo num paradigma de jogos RTS (Secção 2.3). Combinando estes factores, prevê-se possível a redução do *workload* geral do supervisor.

3.8 Base de Desenvolvimento

Uma vez analisada a arquitectura da ferramenta, a forma como todo o software interage, o método de operação das missões com UAV's e os requisitos principais do projecto, é possível determinar com toda a certeza os componentes já existentes no Neptus, que poderão servir de base para a construção do módulo de operação. Como se analisa na Figura 3.12, de todos os plugins existentes no Neptus, visíveis na Figura 3.2, aquele que contém elementos reutilizáveis para a construção do módulo de operações é o *plugin* de *Planning*, mais precisamente a classe *PlanningPanel*. De igual forma será possível utilizar o *SimpleSubPanel*, retirado o código central do Neptus. No entanto foi identificada a necessidade de efectuar algumas alterações ao código basilar do Neptus, neste caso a classe *StateRenderer2D*.

Apesar do limite imposto à alteração de código base do Neptus, identificaram-se algumas modificações absolutamente fundamentais para que fosse possível o correcto funcionamento da consola. Estas modificações foram supervisionadas e aceites na íntegra por elementos da equipa Neptus e consideraram-se o menos intrusivas possível.

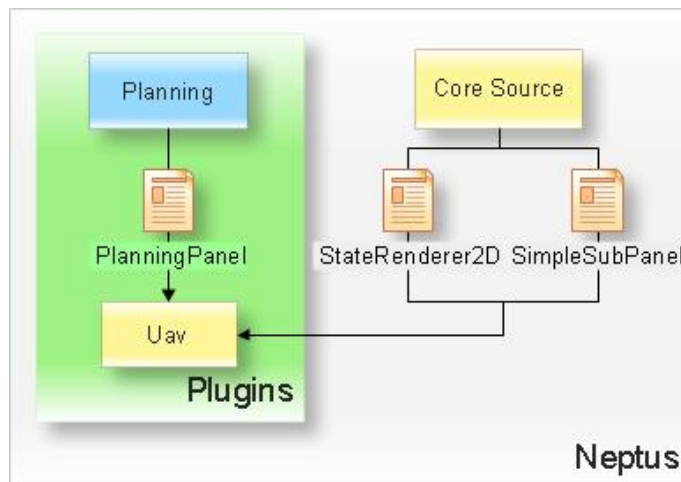


Figura 3.12: Representação dos elementos aproveitados, ou alterados, na implementação do módulo de operação

3.9 Resumo

Neste capítulo foi dada uma pequena introdução à ferramenta que será a base para o desenvolvimento do módulo de operações proposto. Foi feita referência à sua arquitetura base, alertando para certas condições impostas para o desenvolvimento do módulo; foi esquematizada a forma como as várias aplicações de software interagem entre si no decorrer de uma missão; foi, igualmente, apresentado o modo como está organizado o lado humano de uma missão com UAV's e, seguidamente, foram apresentados os requisitos funcionais e não funcionais do projecto, bem como metodologias para abordá-los. Foi explicado o ponto de partida para a fase de implementação, assim como as considerações tomadas antes do início da criação da consola.

De seguida, são apresentados os passos tomados durante a execução do projecto.

Capítulo 4

Implementação

Depois de vistos os pormenores referentes à arquitectura interna do Neptus e ao típico modo de operação de uma missão com UAV's, é possível agora apresentar os detalhes de implementação do módulo de operação concebido.

4.1 Arquitectura do plugin UAV

Antes de avançar com a descrição dos detalhes de cada componente do novo módulo de operações, é necessário apresentar a arquitectura do *plugin* concebido propositadamente para o módulo de controlo, de forma a melhor compreender como decorre a interacção entre todas as partes constituintes. Como ilustra a Figura 4.1, o *plugin* Uav é subdividido em três componentes específicos: *panels*; *painters* e *listeners*. Existindo ainda duas classes - *MapIcon* e *HoveringButton* - partilhadas por estes três componentes. Observêmos então os pormenores de cada um destes constituintes.

4.1.1 Panels

Os *panels* são os elementos base concebidos para o módulo de operação; são seis na sua totalidade. Maioritariamente, estendem o *PlanningPanel* e o *SimpleSubPanel*, como referido na Figura 3.12, e possuem, graças à sua hereditariedade, todos os mecanismos necessários para a sua compatibilidade com os protocolos de comunicação usados durante a missão, bem como os métodos necessários à integração com os *plugins* nativos ao Neptus. Serão os painéis o local onde irão ser disponibilizadas as funcionalidades desejadas pelo supervisor da HRT.

Implementação

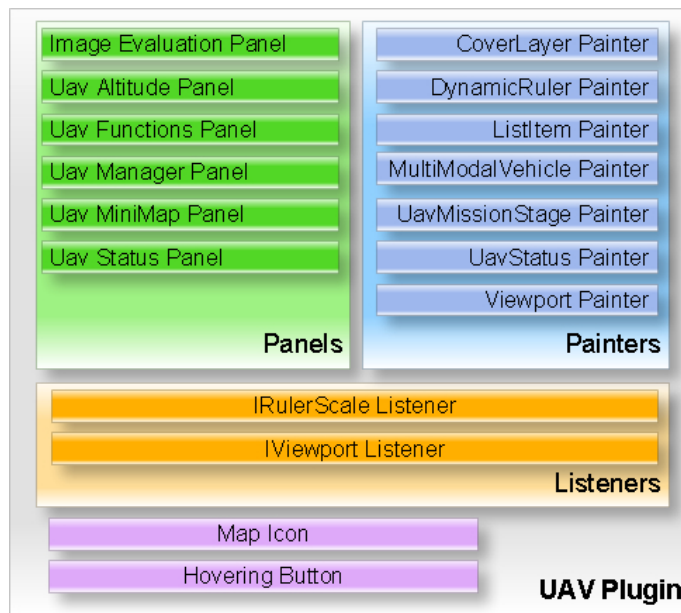


Figura 4.1: Esquema da arquitectura do *plugin* desenvolvido para o Neptus

4.1.2 Painters

Os *painters* surgem como resposta às restrições impostas relativamente à alteração do código base do Neptus. Como referido no Capítulo 3, alterações a código que resida na zona central do Neptus são desencorajadas, pois as repercussões em outras secções de código são difíceis de prever. Considerando que o método principal de desenho para as consolas - *StateRenderer2D* - se encontra dentro desta zona da arquitectura, foi necessário delinear uma forma de contornar este obstáculo. Para esse efeito, são utilizados *painters* externos, que actuam por cima do *renderer* principal. No entanto, é necessário salientar uma consequência importante da não alteração do processo de funcionamento do *StateRenderer2D* - o custo de processamento do método de desenho é o dobro. Isto deve-se ao facto de, ao não ser interrompido ou alterado o fluxo normal do código Neptus já existente, em alguns *panels* há elementos a serem desenhados duas vezes. Esta redundância torna o método de representação mais pesado e ineficiente.

4.1.3 Listeners

No funcionamento normal da consola, existe a frequente necessidade de actualizar e sincronizar informação entre um *panel* e os seus *painters*; desta forma, cada *panel*, caso seja necessário, possui um *listener* associado a cada um dos seus *painters*. Estes *listeners* não são mais do que interfaces que fornecem *templates* de métodos que serão usados para os processos de actualização. Como é visível na Figura 4.2, nem todos os *panels* têm *painters* e, consequentemente, nem todos têm *listeners* associados a si. Apesar

Implementação

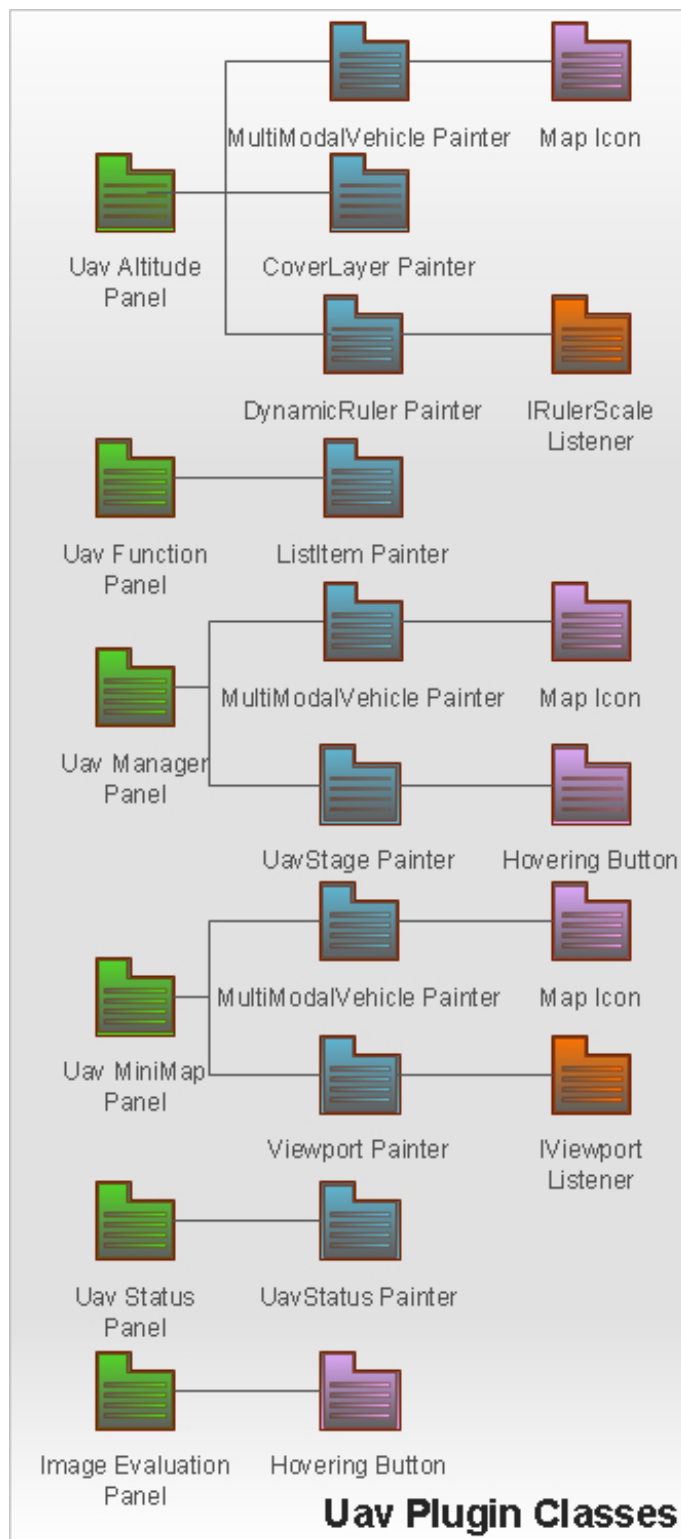


Figura 4.2: Diagrama de classes do *plugin* UAV

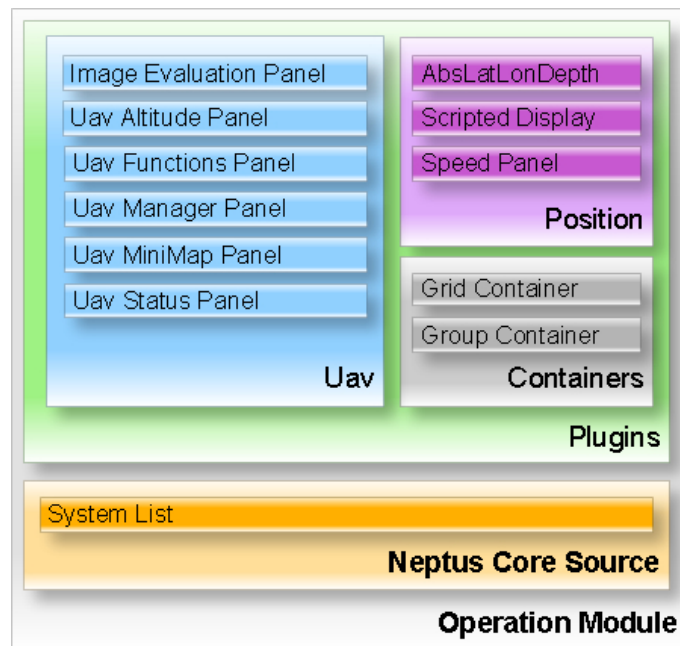


Figura 4.3: Esquema da arquitectura do módulo de operações desenvolvido

dos problemas inerentes ao desempenho e rapidez, esta foi a única solução encontrada que satisfaz as necessidades do projecto e a restrição imposta. Não obstante, continua a manter-se o isolamento entre os métodos dos vários *plugins* e entre estes e o código base do Neptus.

4.2 Arquitectura do Módulo de Operação

Apresentada a arquitectura do *plugin* desenvolvido, segue-se agora a arquitectura do módulo de controlo em si.

Tal como é visível na Figura 4.3, o módulo de operação é composto através da junção de três *plugins*, sendo um deles criado propositadamente para este projecto, com o código base do Neptus. A forma como estes elementos interagem é explicada em detalhe no decorrer deste capítulo. São igualmente expostas a inspiração e as razões para a forma do seu desenvolvimento relativamente ao estudo feito no Capítulo 2 e à análise efectuada no Capítulo 3.

4.2.1 Uav Manager Panel

Este é o painel central de todo o módulo, visível na posição 1 da Figura 4.4, pois permite a visualização dos veículos activos e os seus trajectos durante a missão. Dito isto, é imperativo possibilitar ao supervisor manter a sua atenção maximizada neste componente durante a execução de uma missão. Para isso, são tomadas duas importantes decisões:

Implementação

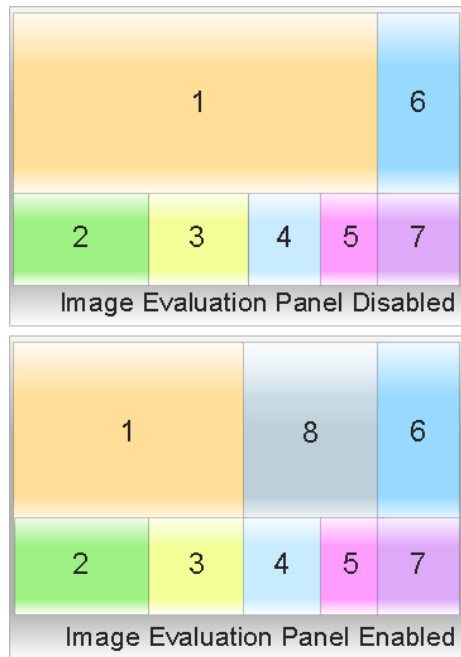


Figura 4.4: Esquema da disposição dos componentes do módulo de operações



Figura 4.5: Protótipo do painel de supervisão principal

- Reservar a maior parte da consola para este componente, como é visível em exemplos retirados de RTS (Figuras 2.5 e 2.6), assegurando assim a atenção do supervisor, de forma a evitar a sua distração com outros componentes.
- Utilizar, como forma de representar os veículos, ícones multi-modais que permitam um maior grau de SA, graças à fusão de vários pedaços de informação numa só representação. Desta forma, assegura-se que o supervisor consiga determinar o grau de atenção que um veículo necessita apenas pela sua representação no ecrã.

Adicionalmente, é possível ao supervisor manter registo da fase de voo actual em que o UAV se encontra, recorrendo a ícones representativos de cada fase, como se verifica na Figura 4.5. A decisão de optar por imagens em detrimento de siglas ou palavras assenta na base de que as representações por imagens são mais claras e simples de interpretar (baseado no estudo presente na Seccção 2.4.2), o que reduz o "ruído" de informação existente quando o supervisor necessita de interpretar e correlacionar grandes quantidades de informação em situações de voo mais críticas. Seguindo o exemplo dado pelo paradigma de jogos RTS, é possível alternar entre os vários UAV's activos, bastando para isso seleccioná-los com o rato; esta *feature* não estava até então disponível no Neptus.

4.2.2 Uav Mini Map Panel

O *Mini Map Panel* (posição 2 da Figura 4.4) funciona como par do *Uav Manager Panel*, sendo que está configurado a um nível de zoom superior ao do seu "irmão". Nesta relação de um para um, o mini mapa fornece um ponto de referência estático do campo de operações, permitindo ao supervisor estar a par dos estados e posições de UAV's activos que não estejam representados no *Uav Manager Panel* devido ao seu nível de zoom. Desta forma, é possível ter um maior grau de vigilância e SA sobre áreas de missão vastas, ao mesmo tempo que viabiliza o aumento, mais seguro, do tamanho das áreas de missão actuais.

Tal como no *Uav Manager Panel*, também neste painel é repetida a utilização de ícones que permitem alertar o supervisor para o grau de atenção que o UAV necessita, sem que este tenha de efectuar repetidas leituras dos dados numéricos do estado no UAV. Para manter o supervisor informado relativamente à posição do campo de visão do *Uav Manager Panel*, é desenhado, sobre o mini mapa, um *viewport*, visível na Figura 4.6, que representa a secção da área de missão que está actualmente a ser visionada através do *Uav Manager Panel*.

O mini mapa é utilizado recorrentemente em jogos RTS como um mecanismo para permitir que o jogador tenha sempre a noção, não só da posição dos elementos que controla, mas também do seu posicionamento no campo de batalha. É exactamente este sentimento de controlo que se pretende atingir com este componente no módulo de operação.



Figura 4.6: Protótipo do mini mapa usado em conjunção com o painel de supervisão principal

4.2.3 Uav Altitude Panel

Como é possível observar na Figura 4.7, o *Uav Altitude Panel* (posição 6 da Figura 4.4), tal como o nome indica, permite ao supervisor manter-se a par, continuamente, da altitude de todos os UAV's activos na área de missão. Este painel vem colmatar uma necessidade sentida pela equipa AsasF e que a ferramenta Neptus previamente não satisfazia. A forma até então encontrada para obter esta informação centrava-se no uso de vários painéis adicionais, não concebidos para essa funcionalidade, o que complicava o esforço do supervisor durante uma missão, pois sem esta informação disponível de forma simplificada e contínua, tornava-se árduo acompanhar as flutuações das altitudes dos UAV's, quer devido a avarias mecânicas, quer devido à normal execução de ordens.

Com este componente, o supervisor consegue ter uma rápida noção do distanciamento vertical entre cada UAV sem ser necessário uma constante rotação entre vários outros painéis. Isto reduz o tempo gasto pelo supervisor para obter e confirmar estes dados, possibilitando então que este se mantenha mais tempo focado no painel central de operações.

Também neste painel é seguido o exemplo de painéis passados, sendo o veículo representado por um ícone elucidativo do grau de atenção que é necessário fornecer ao UAV em questão.

4.2.4 Uav Status Panel

Este painel (posição 4 da Figura 4.4), estando à escuta dos servos de cada secção do UAV, permite uma rápida detecção de falhas localizadas em diferentes partes do veículo.

Implementação

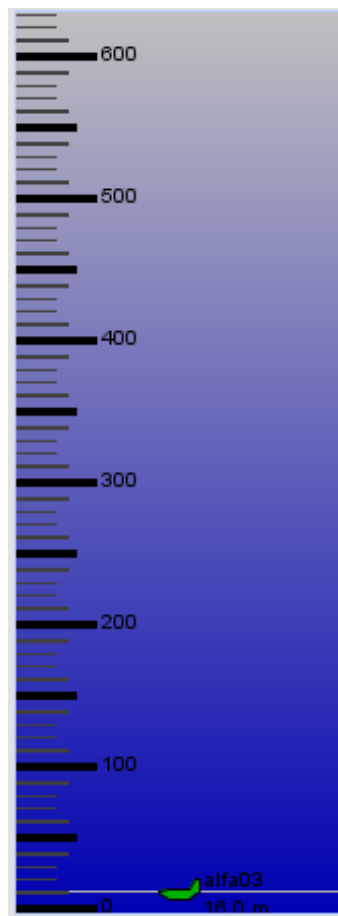


Figura 4.7: Protótipo do painel utilizado para monitorizar a altitude dos UAV's

Este painel torna-se assim uma visão mais detalhada do estado do UAV que permite ao supervisor rapidamente emitir ordens ou correcções a serem feitas pelos operadores ou pilotos, caso as falhas sejam demasiadas. Novamente, são usados princípios descritos na Secção 2.4.2, para maximizar a SA do supervisor e diminuir o seu *workload*.

A essência deste elemento é, não só aumentar a rapidez de diagnóstico de um problema ou falha num UAV graças à veloz identificação de secções problemáticas, mas também diminuir o tempo de resposta do supervisor em situações de emergência que advenham destas leituras. O mesmo nível de rapidez não é atingível se o supervisor tiver de extrapolar falhas de várias fontes de dados numéricos.

Como representa a Figura 4.8, o UAV está subdividido em oito secções que possuem um estado binário - Ok ou Falha - e uma secção, a central, que pode albergar mais ícones representativos do estado dos vários componentes que o UAV leva consigo (bateria, combustível, sensores, etc.).

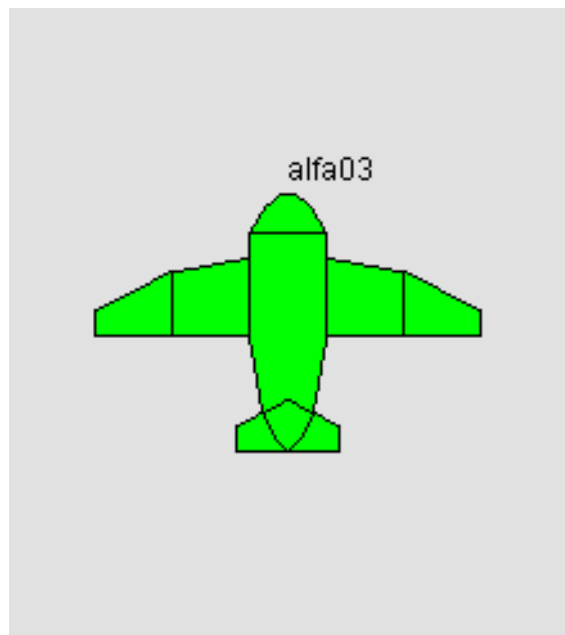


Figura 4.8: Protótipo do painel representativo do estado das várias secções do UAV

4.2.5 Uav Function Panel

O *Uav Function Panel* (posição 5 da Figura 4.4) é concebido como um painel multiusos que pode servir para activar outros painéis que tenham tempos de utilização não permanentes, como é o caso do *Image Evaluation Panel*. Actualmente, tem apenas como utilidade principal a activação do painel de avaliação de imagens e o da consequente listagem das imagens recolhidas até ao momento.

Esta listagem é feita de forma simples - Figura 4.9- e é precedida de um ícone ilustrativo da classificação atribuída, até ao momento, à imagem recolhida.

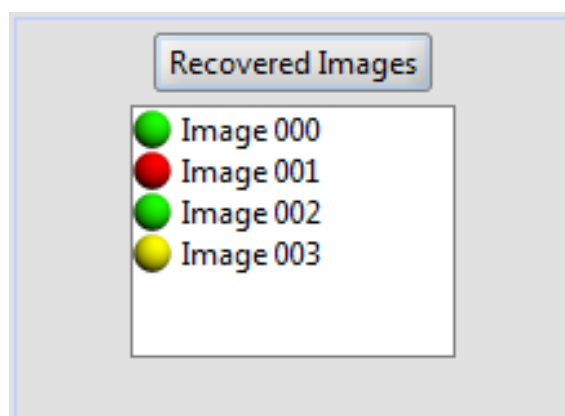


Figura 4.9: Protótipo do painel multiusos da consola

4.2.6 System List

Este componente, visível na posição 7 da Figura 4.4, já é nativo ao Neptus e, tal como é possível ver na Figura 4.10, permite ao supervisor detectar se foi perdida a ligação com a consola PCC que opera o seu UAV respectivo. Para além disso, pode-se alternar entre os vários veículos activos, bastando para isso seleccionar o seu botão respectivo.



Figura 4.10: Imagem do painel Neptus que lista os veículos activos numa consola

Com este painel, é possível rapidamente aceder a veículos que não estejam dentro do campo de visão do *Uav Manager Panel*. É um método alternativo ao de selecção directa no *Uav Manager Panel*.

4.2.7 Image Evaluation Panel

Este elemento (posição 8 da Figura 4.4) destina-se à avaliação *on the fly* de imagens recolhidas pela esquadra de UAV's que esteja no ar. O painel é responsável por, periodicamente, verificar o directório que aloja as fotos recolhidas e adicioná-las à listagem presente no *Uav Function Panel*. As classificações atribuídas são guardadas num ficheiro XML, igualmente responsável por armazenar a identificação do veículo que produziu a imagem.

Durante o desenvolvimento do painel, foi preocupação constante, tal como com o resto dos elementos produzidos, manter o método de operação o mais simples possível, de forma a permitir que o supervisor consiga desempenhar esta função sem que isso reduza

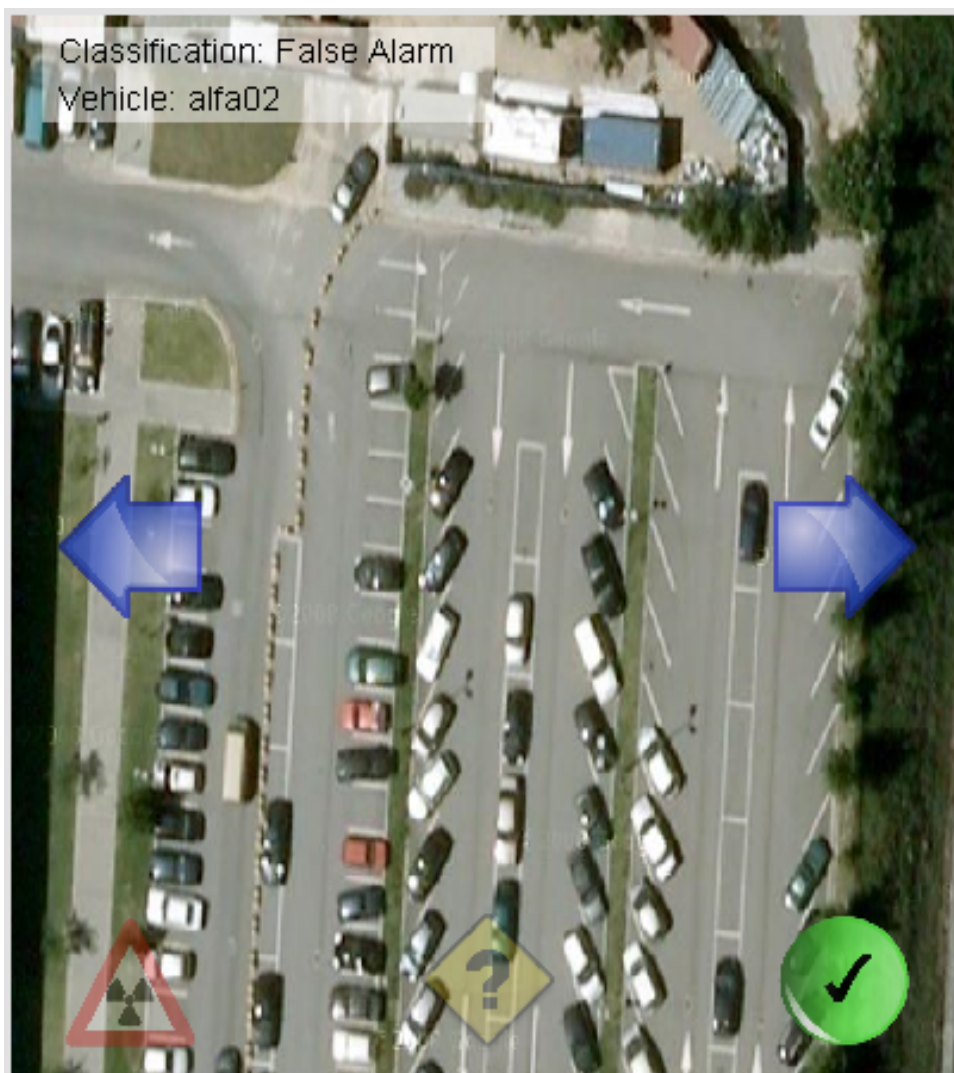


Figura 4.11: Protótipo do painel de avaliação de imagens

em demasia a sua SA. Assim sendo, o método de classificação é efectuado através de três ícons representativos dos três veredictos possíveis - *Threat*, *Inconclusive*, *False Alarm*. Na criação destes ícons foi tomada atenção a três variáveis que permitem a identificação rápida e concisa dos seus significados e funções:

- **Forma** - Como é possível ver, analisando a Figura 4.13, no nosso dia-a-dia, utilizamos a forma dos objectos como um poderoso meio de alertar e informar acerca de situações que nos rodeiam. É precisamente devido a esse hábito que adoptamos a utilização de formatos, já conhecidos no dia-a-dia, para moldar os ícons de avaliação do painel;
- **Cor** - Olhando para a Figura 4.12, é fácil concluir como, tal como a forma, a cor é utilizada como meio de transmitir informação de forma simples e eficiente.



Figura 4.12: Exemplo do uso de cores para transmitir informação

Do mesmo modo, os ícons de classificação do *Image Evaluation Panel* utilizam a mesma simbologia, de forma a advertir, rapidamente, qual a classificação da imagem que este está a ver;

- **Ícon Central** - Finalmente, são utilizadas imagens que tipificam as situações que representam, desde o perigo à confirmação, passando pela dúvida.

Desta forma, conseguimos conceber uma interface que se apoia quase exclusivamente em reflexos condicionados. Isto possibilita um controlo, por parte do supervisor, muito mais intuitivo.

4.2.8 Generic Displays

Por fim, é necessário referir os painéis genéricos (posição 3 da Figura 4.4), já presentes nos *plugins* do Neptus, que são utilizados para apresentar, em formato de texto, dados relativos ao estado e posicionamento do UAV. Os dados presentes na Figura 4.14



Figura 4.13: Exemplo de formato de ícon para transmitir informação

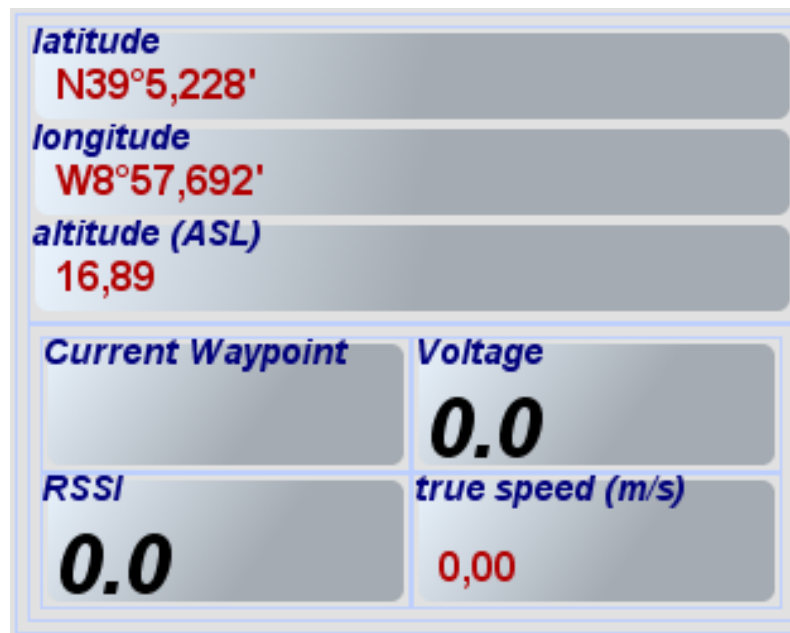


Figura 4.14: Imagem exemplo da utilização de *displays* genéricos existentes no Neptus

são apenas um exemplo do que é possível aceder através destes painéis. Graças à forma independente como foram desenvolvidos, estes painéis podem ser adicionados ou retirados à consola sem que esta pare a sua execução. Cada *display* é configurado por um simples *script* de JavaScript que pode ser introduzido ou alterado, caso seja detectada essa necessidade no meio de uma missão.

4.3 Cumprimento de Requisitos Funcionais

É agora necessário averiguar o cumprimento dos requisitos impostos na secção 3.7.2, relativamente às funcionalidades a serem implementadas.

Como se observa na Tabela 4.1, todos os requisitos considerados de elevada prioridade foram implementados, bem como um dos requisitos considerado de prioridade média. No entanto, por questões de tempo ou complexidade de implementação, foram deixados de parte dois requisitos, sendo um deles o que inicialmente tinha menor prioridade.

Tabela 4.1: Tabela de cumprimento dos requisitos funcionais

Requisito	Prioridade	Implementado
Emitir Ordens	Baixo	Não
Visualizar vários UAV's	Alta	Sim
Visualizar dados de UAV	Alta	Sim
Receber Alarmes	Média	Não
Avaliar Images Recebidas de UAV	Média	Sim

Relativamente ao cumprimento dos requisitos não funcionais, esses só conseguirão ser averiguados na secção de Resultados deste documento, visto ser necessário input dos utilizadores da consola.

4.4 Resumo

Neste capítulo, foi apresentada a arquitectura do *plugin* desenvolvido propositadamente para o módulo de operações, juntamente com a arquitectura do módulo em si. Foi, adicionalmente, feita uma listagem de todos os componentes do módulo e os detalhes por detrás do seu desenvolvimento. Finalmente, foi dado um ponto de situação relativamente ao cumprimento dos requisitos funcionais do projecto.

Capítulo 5

Resultados

Feita a descrição da implementação do módulo de operações, é agora essencial apresentar o conjunto de testes e seus resultados, efectuados de modo a determinar o cumprimento dos requisitos não funcionais do projecto. Todos os detalhes relativos aos testes efectuados podem ser consultados no Anexo [A](#).

5.1 Testes Efectuados

De forma a determinar a aceitação do novo módulo de operações, foram efectuados dois simulacros, em ambiente controlado no LSTS, em que cada um tinha um diferente número de UAV's, condições e objectivos. De igual forma, cada simulacro teve um supervisor diferente, com o objectivo de alargar o espectro de inputs recebidos. Os elementos que participaram nas simulações são certificados em supervisão de missões e em operação de UAV's. Como supervisores, foram utilizados os dois supervisores sénior dos AsasF, já experientes em inúmeras missões reais onde desempenham precisamente esse papel.

5.1.1 Workload

Um dos factores principais para determinar a aceitação do módulo de operações é a sua capacidade em reduzir o *workload* que o supervisor sente no decorrer de uma missão. Tal como referido no Capítulo [2](#), Secção [2.4.3](#), de forma a avaliar o factor do *workload*, foi utilizada a ferramenta NASA-TLX. Após cada um dos simulacros, foi efectuado um questionário ao supervisor, relativamente à sua experiência com o módulo, sendo esses dados acessíveis, na sua íntegra, no Anexo [A](#). Ambos os simulacros foram efectuados utilizando tanto o módulo actualmente à disposição dos AsasF, como o novo módulo de operações.

Resultados

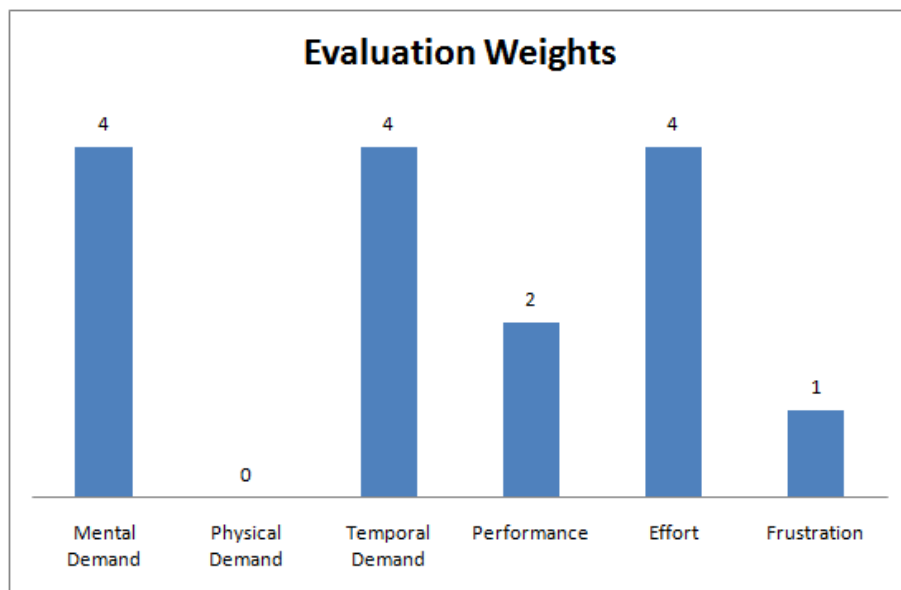


Figura 5.1: Gráfico dos diferentes pesos atribuídos às componentes da ferramenta NASA-TLX no Simulacro 01

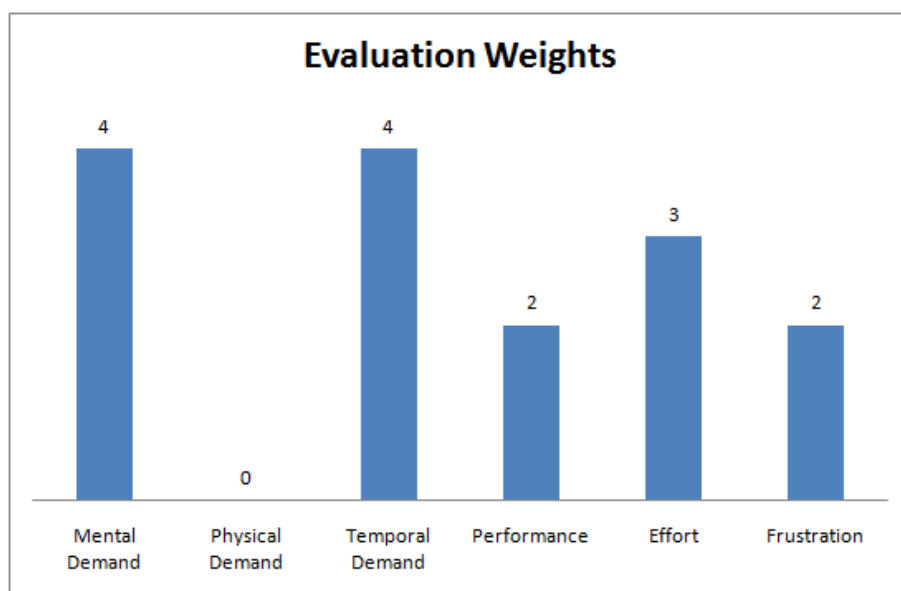


Figura 5.2: Gráfico dos diferentes pesos atribuídos às componentes da ferramenta NASA-TLX no Simulacro 02

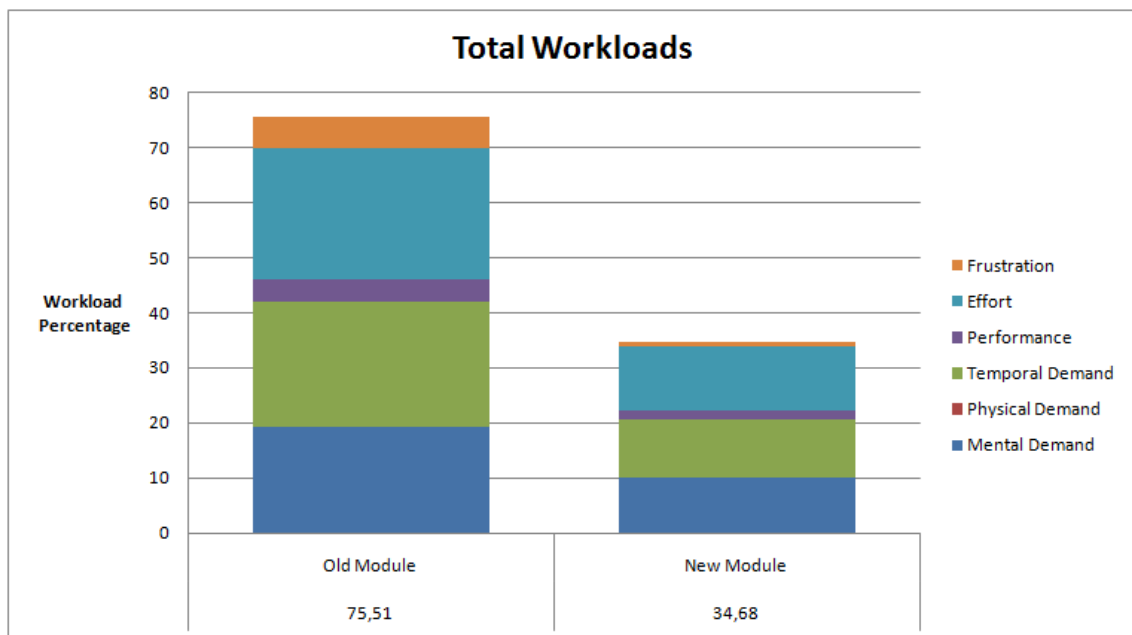


Figura 5.3: Comparação do *workload* total entre os diferentes módulos de operação no Simulacro 01

No decorrer dos testes, e tal como dita o protocolo de utilização da ferramenta NASA-TLX, foram recolhidos os pesos relativos aos seis componentes de *workload* avaliados por este método. É interessante verificar - Figura 5.1 e Figura 5.2 - que os componentes relativos à exigência mental, exigência temporal e esforço são, de longe, considerados os mais relevantes na avaliação do *workload* para a tarefa desempenhada pelo supervisor. Isto leva-nos a concluir que reduções nestes componentes afectarão em larga escala o *workload* total da tarefa.

Efectuada a recolha, análise e cálculo dos valores de *workload* para cada componente, é possível fazer uma comparação entre ambos os módulos de operação. Como se pode verificar pela observação das Figuras 5.3 e 5.4, foi registada uma descida abrupta do *workload* da tarefa do supervisor, do módulo antigo para o módulo novo. Ao registar uma descida tão íngreme, pode-se extrapolar que as componentes que o supervisor catalogou como as mais significativas para o *workload* da tarefa desempenhada sofreram variações elevadas.

Observando as Figuras 5.5 e 5.6, podemos verificar que em todas as componentes de *workload* foi registada uma redução. No entanto, prestando atenção ao gráfico, detectamos que as enormes reduções se verificam, de facto, nos componentes de exigência mental, exigência temporal e esforço. Isto possibilita-nos concluir que o novo módulo de controlo veio responder, de forma extremamente eficaz, às necessidades sentidas pela equipa AsasF em relação ao módulo de operações antigo, visto ser possível registar reduções acima dos 50% (Figuras 5.7 e 5.8) nos componentes cotados como os mais cruciais

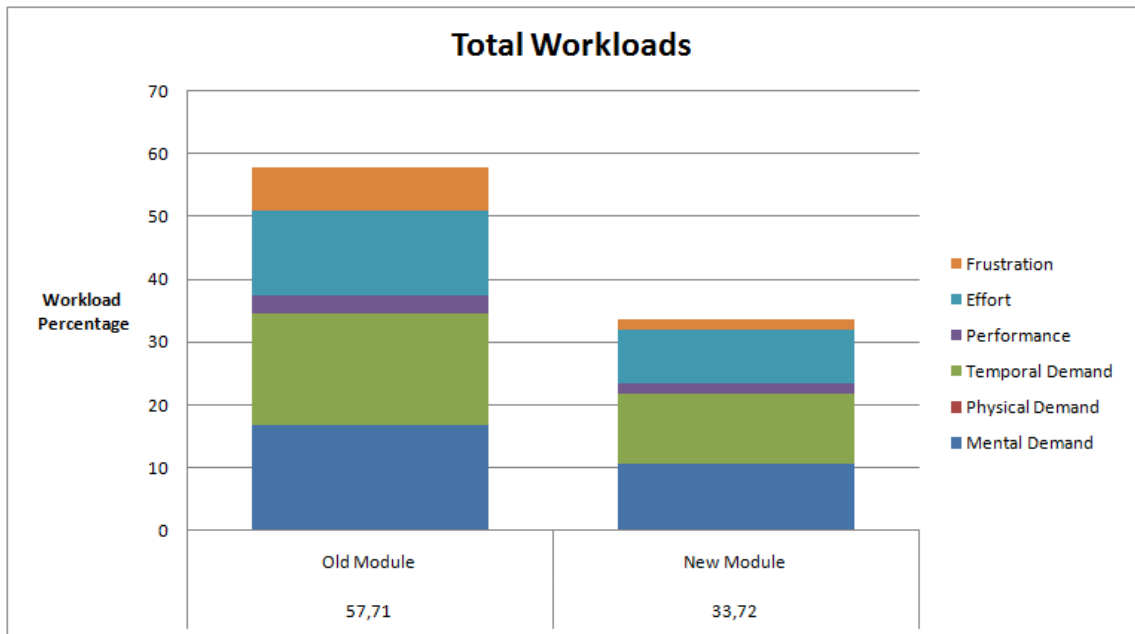


Figura 5.4: Comparação do *workload* total entre os diferentes módulos de operação no Simulacro 02

para o *workload*.

5.1.2 Situation Awareness

Avançando para a análise da SA, e tal como referido anteriormente no Capítulo 2, optou-se pela utilização do método SAGAT para a recolha destes dados, que podem ser consultados na íntegra no Anexo A, Secção A.1.5. Os detalhes relativos ao método utilizado e ao teor das questões podem ser analisados no Anexo C.

A confirmação do aumento da SA é um requisito importante para o sucesso do módulo, sendo este um requisito de alta prioridade, à semelhança da redução da *workload*. Como é representado na Figura 5.9, existem claras subidas de SA entre o módulo antigo e o novo, apesar de em dois pontos existir uma pequena descida. No entanto, comparando a quantidade de subidas (Figuras 5.9 e 5.10) com a quantidade de descidas, bem como o valor percentual dessa variação (Figuras 5.11 e 5.12), é possível concluir que, no geral, a SA do supervisor, durante a sua tarefa, aumentou.

5.1.3 Fan-Out

Apesar da sua importância como métrica em HRT's, visível no Capítulo 2, concluímos que, dentro do ambiente em que decorreram os simulacros, seria demasiado complicado determinar com precisão qualquer valor de FO. As razões que levaram a esta conclusão foram as seguintes:

Resultados

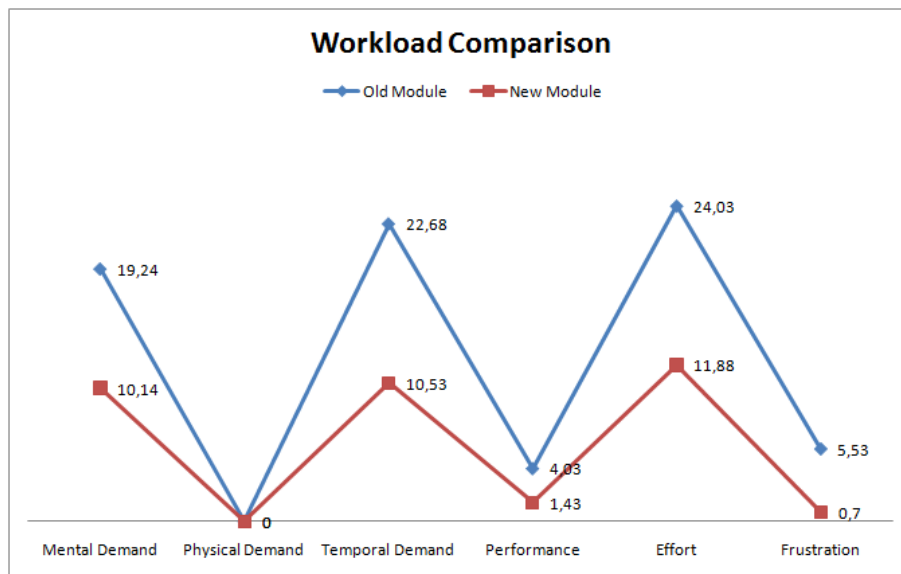


Figura 5.5: Gráfico apresentando a comparação do *workload*, dividido nas suas componentes, entre os diferentes módulos de operação no Simulacro 01

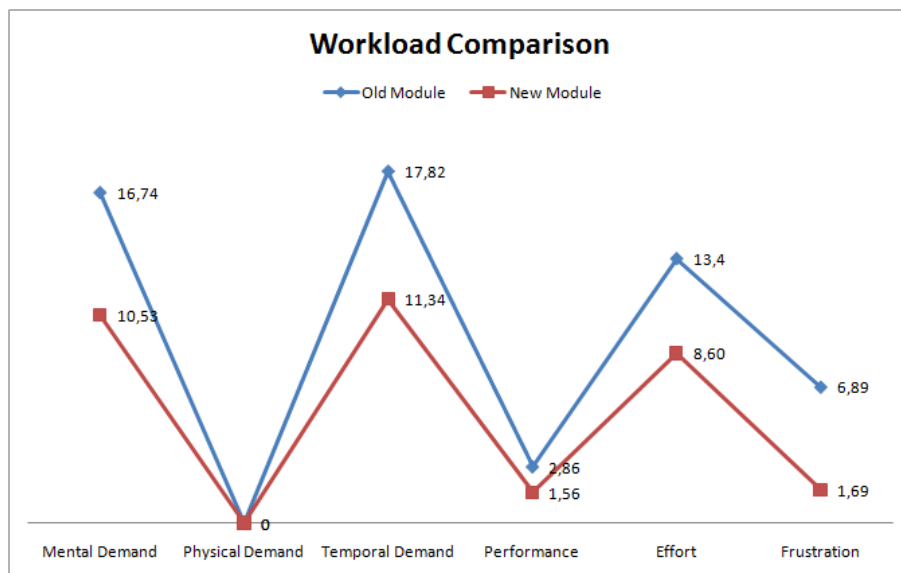


Figura 5.6: Gráfico apresentando a comparação do *workload*, dividido nas suas componentes, entre os diferentes módulos de operação no Simulacro 02

Resultados

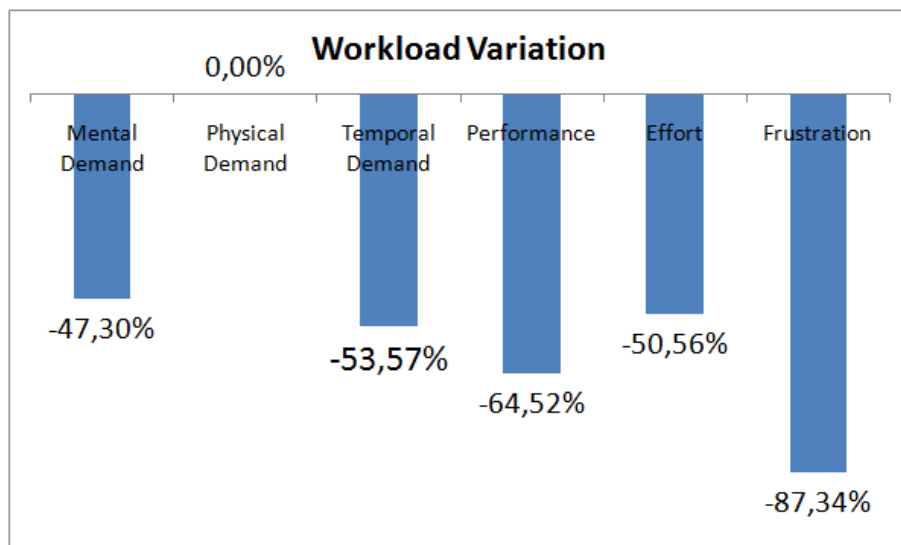


Figura 5.7: Apresentação da variação percentual do *workload* registrada para cada componente no Simulacro 01

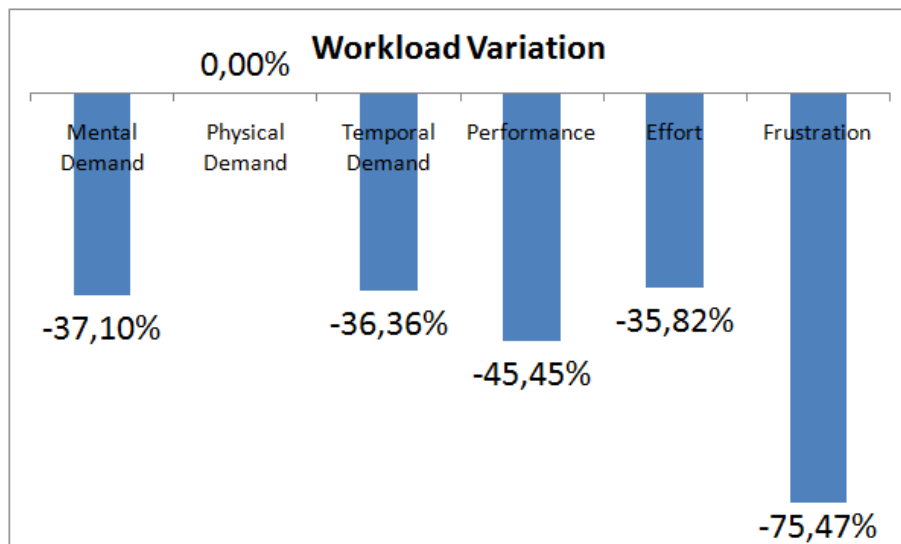


Figura 5.8: Apresentação da variação percentual do *workload* registrada para cada componente no Simulacro 02

Resultados

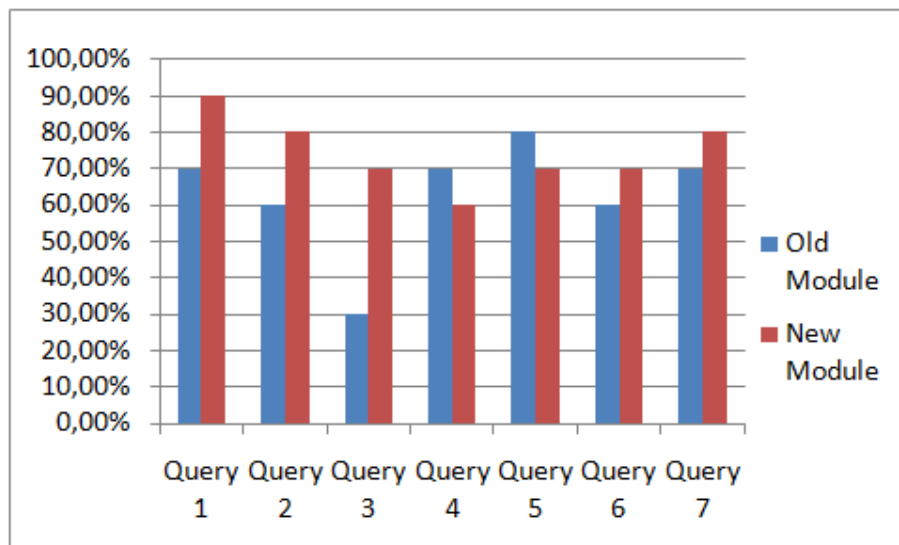


Figura 5.9: Representação das diferenças de SA entre ambos os módulos no Simulacro 01

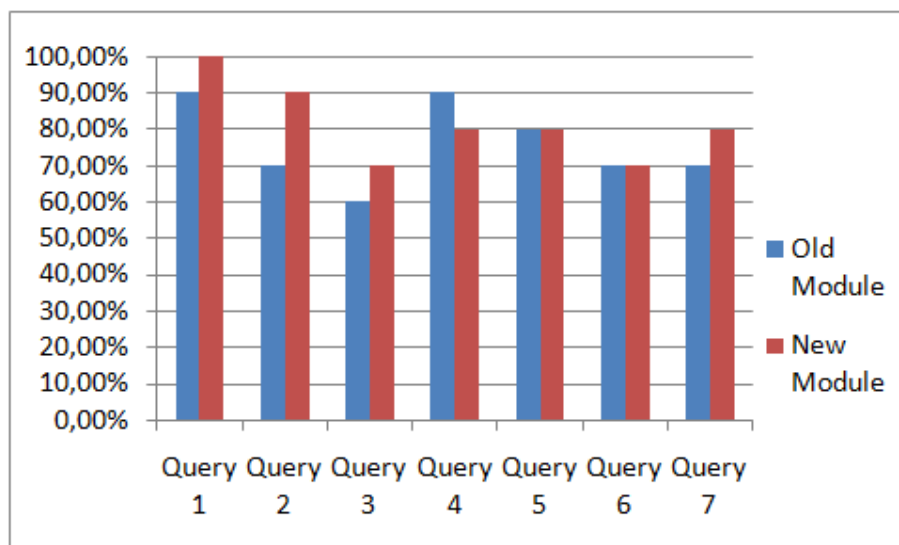


Figura 5.10: Representação das diferenças de SA entre ambos os módulos no Simulacro 02

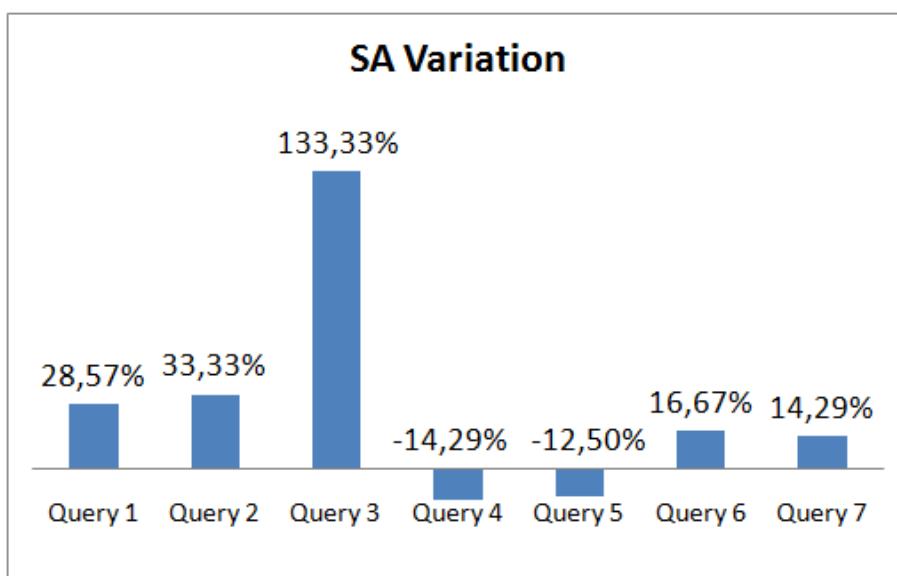


Figura 5.11: Gráfico da variação percentual do SA entre ambos os módulos no Simulacro 01

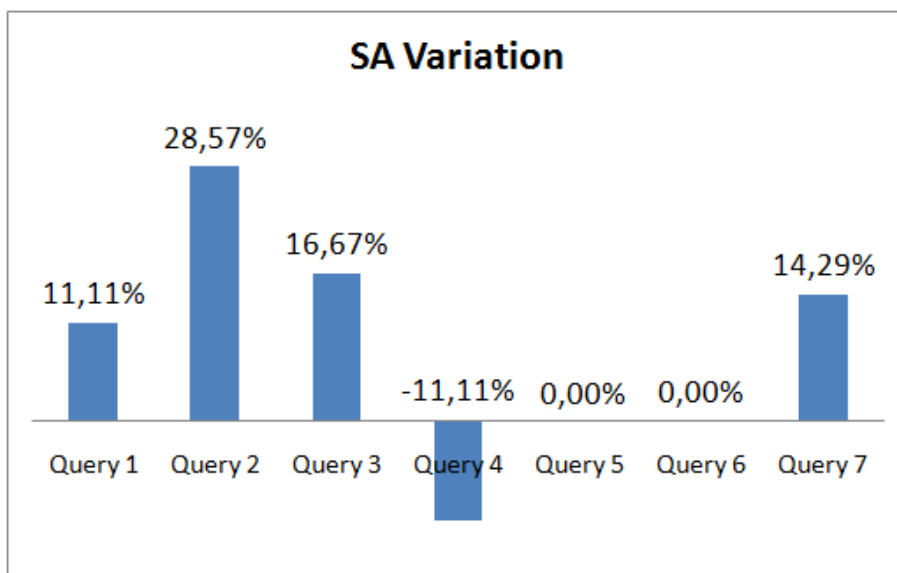


Figura 5.12: Gráfico da variação percentual do SA entre ambos os módulos no Simulacro 02

- **Número limitado de veículos** - Apesar da versatilidade de simulação permitida pelo Neptus, o número de veículos que é possível simular em simultâneo numa missão é limitado pelo número máximo de operadores humanos presentes na equipa e pelo número de veículos configurados para serem reconhecidos pelo Neptus. Entre estes dois valores, o que limita mais a escala do simulacro é o número de operadores humanos, uma vez que é necessário um operador para cada avião simulado. Caso não fosse um pré-requisito para o operador possuir prática com as consolas PCC, seria então possível aumentar este valor de três UAV's máximos para um que permitisse testar mais a fundo a ferramenta. De qualquer forma, mesmo que fosse possível aumentar o número de operadores, existiria ainda a necessidade de configurar o Neptus de modo a aceitar novos veículos para simulação. Visto não ser normal o LSTS adquirir regularmente novos UAV's, o processo de adição de novos veículos não está *stream-lined*, sendo por isso um processo demorado e que deve ser feito por responsáveis do Neptus. Tudo isto combinado torna bastante trabalhosa e complexa a tarefa de aumentar então o número máximo veículos presentes em simulação, de forma a determinar com certeza um limite de *fan-out*;
- **Modo de interacção da equipa** - Como já foi referido no Capítulo 3, Seccção 3.6, toda a comunicação entre os vários elementos da missão é desenrolada utilizando software externo VoIP. Aliando isto ao facto do número máximo de veículos simulados ser pequeno e o ambiente de simulação ser controlado, existindo assim pouca interferência ou ruído na comunicação, torna-se quase imperceptível detectar diferenças entre a supervisão de dois ou três UAV's.

Com estas limitações, torna-se bastante complicado recolher dados que confirmem o aumento ou diminuição do FO com este novo módulo de operações, comparativamente ao anterior.

5.1.4 Curva de Aprendizagem

Relativamente à curva de aprendizagem do novo módulo, tomando em consideração o facto de terem sido apenas feitos dois simulacros, ambos com supervisores experientes e, portanto, já bastante familiarizados com o módulo antigo e com algumas partes do novo, determinou-se que não estavam reunidos dados suficientes para averiguar, com segurança, se a curva de aprendizagem do módulo era ou não reduzida. Caso fosse possível executar alguns simulacros com supervisores menos experientes, poderíamos então ver como alguém menos versado na ferramenta se adapta ao novo módulo e, fazendo um cruzamento entre esses dados e os dos supervisores mais experientes, seria igualmente possível concluir se existem diferenças entre as curvas de aprendizagem de ambos os módulos.

Não obstante, é necessário deixar explícito que não foram detectadas ou mencionadas nenhuma dificuldades durante a execução dos simulacros. Pelo contrário, foram tecidos

comentários alusivos à simplicidade da interface do módulo criado e à facilidade da sua utilização. No entanto, esses reparos foram considerados insuficientes para constituírem um retrato fiel da facilidade de aprendizagem do módulo.

5.2 Cumprimento de Requisitos Não Funcionais

Efectuada a análise dos resultados recolhidos, é possível então concluir que, dos quatro requisitos não funcionais impostos - Figura 5.1 -, apenas os dois requisitos com menor prioridade não puderam ser averiguados com segurança.

Tabela 5.1: Tabela de cumprimento dos requisitos não funcionais

Requisito	Prioridade	Cumprido
Reduzir <i>workload</i> do supervisor	Alto	Sim
Aumentar a SA do supervisor	Alta	Sim
Aumentar o FO de uma missão	Média	Não
Manter a curva de aprendizagem para o módulo reduzida	Baixa	Não

5.3 Análise aos Componentes do Módulo

Além de determinar o cumprimento dos requisitos não funcionais, torna-se igualmente interessante delimitar os componentes do novo módulo que, aos olhos dos supervisores, tiveram maior influência nos resultados verificados. Nesse sentido, foi efectuada um questionário no final de cada simulacro, de forma a determinar este factor. O questionário refere-se apenas a aspectos do novo módulo e todos os detalhes do mesmo podem ser encontrados no Anexo A, Secção A.1.6.

É interessante verificar que o painel considerado mais útil é o *Uav Altitude Panel*, sendo argumentada a sua adição como uma enorme melhoria ao sistema actual. Apesar da sua aparente simplicidade, este componente fornece, de forma rápida e concisa, informação considerada vital ao supervisor. Este factor, por si só, confere um maior grau de confiança ao supervisor no decorrer da sua tarefa.

Da mesma forma, é curioso detectar que o painel considerado menos essencial à tarefa de supervisão, excluindo o *Image Evaluation Panel*, devido à inexistência de imagens recolhidas nos simulacros, é o *Uav Mini Map Panel*. A argumentação para este facto é a de que, devido ao pequeno tamanho da área de missão, a necessidade de um mini mapa foi pouco sentida. No entanto, a sua utilidade num ambiente maior foi confirmada pelos supervisores inquiridos. Um outro argumento a favor deste painel é o de que a sua inutilidade adveio igualmente da experiência pessoal dos supervisores, querendo com isto dizer que a familiaridade que estes têm com a área de missão pode ter levado a que, durante o decorrer da missão, não se terem apercebido da existência do painel. Apesar

disso, com a aparição de novos supervisores, assim como com o aumento das áreas de missão, este painel poderá ganhar mais ênfase durante o cenário de missão.

Finalmente, a abordagem utilizada para a representação dos veículos, e a inclusão de ícons representativos da fase actual de voo, foram vistas como bons complementos ao módulo pois permitem, por um lado, apresentar informação que até ao momento não era disponibilizada em local algum e, por outro lado, facilitar a gestão da atenção que o supervisor tem de dividir por todos os UAV's activos.

Naturalmente, este questionário permite o levantamento de algumas melhorias que podem ainda ser incorporadas no módulo de operações, mas relativamente a elas é feita uma apresentação mais detalhada no Capítulo 6.

5.4 Resumo

Nesta secção do documento foram apresentados os resultados obtidos após a sequência de testes comparativos efectuada a ambos os módulos de operações. Foi possível demonstrar assim uma análise do cumprimento dos requisitos não funcionais do projecto, bem como a descoberta dos principais componentes do novo módulo que contribuem para os resultados obtidos.

Seguidamente, são apresentadas as conclusões do projecto da dissertação.

Capítulo 6

Conclusões e Trabalho Futuro

Finalmente, atingimos a secção de conclusões, onde será feito um remate da situação final do projecto, bem como o delineamento de tópicos para trabalho futuro.

6.1 Resumo do Trabalho Efectuado

O objectivo fundamental desta dissertação centrava-se na criação de um novo módulo de operações para a equipa AsasF do LSTS, que facilitasse as funções do supervisor responsável por uma equipa de UAV's. Este módulo teria de ser concebido a partir da ferramenta Neptus, actualmente utilizada pelo LSTS, e deveria ser utilizado como base de concepção um paradigma baseado em jogos RTS. De forma a alcançar o objectivo proposto, foram seguidos os seguintes passos:

- Realizou-se um levantamento de sistemas actualmente utilizados no controlo e supervisão de equipas de UV's, de forma a determinar métricas que permitissem avaliar a existência de progressos entre o módulo actualmente usado pelos AsasF e o módulo criado. Desse levantamento, emergiram as métricas do *workload*, *fan-out* e da *situational awareness*;
- Seguidamente, foi efectuada uma pesquisa relativamente ao uso do paradigma de jogos RTS, como base de concepção para módulos de controlo e supervisão de equipas de UV's. Desta pesquisa, surgiram diversos exemplos que nos permitiram concluir que este conceito trouxe resultados positivos, primariamente a factores de usabilidade;
- Adicionalmente, pesquisaram-se ferramentas e métodos de avaliar com sucesso as métricas determinadas. Concluimos que o uso de SAGAT, para determinação de SA, e NASA-TLX, para determinação do *workload*, seriam satisfatórios;

- Foi efectuado um levantamento do estado actual do projecto, bem como dos protocolos de missão dos AsasF de forma a melhor modelar o projecto. Foram, igualmente, determinados os requisitos de aceitação do módulo de operações, bem como o seu ponto inicial de desenvolvimento;
- De seguida, desenvolveu-se um protótipo do módulo de operações. Este protótipo, aquando da sua criação, foi obrigado a respeitar rígidas condições de desenvolvimento que condicionavam a alteração de código Neptus já existente. De igual modo, foi necessário conceber o módulo em forma de *plugin*, para que o Neptus funcionasse de forma completamente independente à sua existência. Este protótipo foi sujeito a testes repetidos, durante a sua criação, e a duas simulações finais com o objectivo de averiguar a sua prontidão para situações realistas. Estes simulacros colocaram, nas mesmas situações de missão, o módulo actual contra o módulo novo;
- Finalmente, foram feitas análises às métricas de *workload*, *fan-out* e *situational awareness*, para poder determinar o cumprimento dos objectivos propostos para a dissertação.

6.2 Satisfação dos Objectivos

Comparando os objectivos iniciais do projecto com a análise dos resultados recolhidos, durante os simulacros, podemos concluir que todos os objectivos, assinalados com prioridade alta, foram alcançados. Dos objectivos de prioridade média, apenas um em três foi atingido; nenhum dos dois objectivos de prioridade baixa foi cumprido. No entanto, é possível assinalar que o protótipo oferece novas funcionalidades que colmatam importantes necessidades, presentes no LSTS, e consegue, ao mesmo tempo, atingir uma redução de *workload* e um aumento de *situation awareness* que viabiliza a utilização exclusiva deste módulo para missões efectuadas pelos AsasF. De igual modo, foi possível confirmar os benefícios que um paradigma, baseado em jogos RTS, trouxe à concepção de um módulo de operações para supervisão de uma equipa de UAV's.

O módulo está, presentemente, integrado dentro da *framework* Neptus e pronto a usar de forma preliminar no terreno.

O trabalho desenvolvido nesta dissertação trouxe, como contributo, a inclusão dos seus resultados e dos detalhes de implementação no artigo "*Authority sharing in mixed initiative control of Multiple Uninhabited Aerial Vehicles*", aceite para apresentação na sessão intitulada "*Cognitive Design for Human-Automation Interaction in Vehicle Guidance*" e para publicação nas *conference proceedings* da HCI International 2011 [art].

6.3 Trabalho Futuro

No entanto, e apesar ter sido concretizada a concepção e implementação de um protótipo funcional que cumpre uma larga percentagem dos objectivos propostos, é ainda possível determinar melhorias a acrescentar ao módulo, para que este se torne mais maduro e mais robusto para testes em ambiente real. Neste tema, podemos delinear dois grupos de implementações futuras - as que são exequíveis a curto prazo e outras, que podem ser executadas a longo prazo.

A curto prazo teríamos:

- **Efectuar as trocas de fase de voo, no módulo de operações, automaticamente** - Como detectado nos simulacros, a possibilidade de registar a fase actual do voo do UAV tinha um uso reduzido pois, normalmente, o supervisor esquecia-se de alterar o seu estado. Apesar de ter sido reconhecido grande valor a esta funcionalidade, foi possível verificar, na prática, que na ausência de um protocolo de missão que obrigue ao seu uso, esta funcionalidade permanecia subaproveitada. No entanto, caso a troca fosse feita de modo automático, seria então possível ao supervisor tirar o máximo partido desta capacidade do módulo;
- **Possibilitar a emissão directa de ordens entre o módulo de operações do supervisor e as consolas de operadores Neptus** - Esta seria uma adição interessante pois, por um lado, seria mais um passo em direcção ao abandono do PCC como consola para controlo de UAV e, por outro, permitiria uma salvaguarda ao método actual de emissão de ordens. Ou seja, ao permitir que o supervisor insira directamente na consola as alterações às ordens do UAV, é aliviada a comunicação por TeamSpeak e, caso esta fique indisponível devido a avaria ou "ruído", continua a ser possível gerir a equipa de UAV's em segurança;
- **Integrar o TeamSpeak com o módulo de operações do supervisor** - Recorrendo à API do TeamSpeak, seria uma mais valia permitir ao supervisor determinar qual dos elementos da sua equipa está a pedir indicações de missão. Isto faria com que o supervisor pudesse manter a sua atenção sempre no módulo de operações, sem ter de se dirigir ao software externo, caso tenha necessidade de trocar de canal ou identificar um operador.

A longo prazo teríamos:

- **Adição de agentes para auxiliar o supervisor** - Não sendo este um conceito novo, a verdade é que a introdução de agentes com inteligência artificial no módulo de operações viria complementar os esforços de gestão que o supervisor sente durante uma missão. Estes poderiam ficar encarregues de operações de vigilância do estado dos vários UAV's, emitindo automaticamente ordens de correcção de planos

de voo, caso fosse detectado perigo. Alternativamente, poderia alertar o supervisor, deixando ao critério deste o envio ou não dessas ordens. De qualquer forma, é possível especular que esta medida poderia levar a uma maior redução do *workload* do supervisor;

- **Criação de um plugin de *gaze-tracking*** - A inclusão deste *plugin* permitiria detectar e seguir o olhar do supervisor, o que tornaria possível a criação de um *heat map* dos locais para onde este olha no decorrer normal de uma missão ou em situações de stress. Isto permitiria descobrir, com maior segurança, quais os componentes mais importantes de cada módulo, assim como os que apenas estão a causar "ruído" ao supervisor. Numa maior escala, este *plugin* poderia ser utilizado para efectuar *retro-fitting* de consolas já presentes no Neptus, aumentando assim a credibilidade da ferramenta no seu todo.

Referências

- [Ada02] Julie A. Adams. Critical considerations for human-robot interface development. In *Proceedings of 2002 AAAI Fall Symposium*, pages 1–8, 2002.
- [art] Hci international 2011. <http://www.hcii2011.org/> (visitado a 26 Jan 2011).
- [AT10] Gautreault Fabien Laurent Mickaël Causse Mickaë Dehais Frédéric Aubert Thomas, Corjon Julien. Improving situation awareness of a single human operator interacting with multiple unmanned vehicles: first results. In *HUMOUS'10 conference - HUMans Operating Unmanned Systems*, 2010.
- [ATP95] M J Adams, Y J Tenney e R W Pew. Situation awareness and the cognitive management of complex systems. *Human Factors*, 37(1):85–104, 1995.
- [BB09] Matthew L. Bolton e Ellen J. Bass. Comparing perceptual judgment and subjective measures of spatial awareness. *Applied Ergonomics*, 40(4):597 – 607, 2009.
- [CC07] Jacob W. Crandall e M. L. Cummings. Developing performance metrics for the supervisory control of multiple robots. In *HRI '07: Proceedings of the ACM/IEEE international conference on Human-robot interaction*, pages 33–40, New York, NY, USA, 2007. ACM.
- [CM07] M.L. Cummings e P.J. Mitchell. Operator scheduling strategies in supervisory control of multiple uavs. *Aerospace Science and Technology*, 11(4):339 – 348, 2007. COGIS '06.
- [DGP⁺07] Paulo Sousa Dias, Gil M. Gonçalves, Fernando Lobo Pereira, José Pinto, Rui Gonçalves e João Borges Sousa. Neptus, command and control infrastructure for heterogeneous teams of autonomous vehicles. 04/2007 2007.
- [DN08] Angela DiDomenico e Maury A. Nussbaum. Interactive effects of physical and mental workload on subjective workload assessment. *International Journal of Industrial Ergonomics*, 38(11-12):977 – 983, 2008.
- [DSY03] Jill L. Drury, Jean Scholtz e Holly A. Yanco. Awareness in human-robot interactions. In *Proceedings of the IEEE Conference on Systems, Man and Cybernetics*, pages 111–119, 2003.
- [dun07] Seascout on-board software distribution 0.1, 2007. <http://whale.fe.up.pt/seascout/> (visitado a 26 Jan 2011).

REFERÊNCIAS

- [HSS07] Adam J. Hassell, Philip Smith e David Stratton. An evaluation framework for videogame based tasking of remote vehicles. In *Proceedings of the 4th Australasian conference on Interactive entertainment*, IE '07, pages 10:1–10:8, Melbourne, Australia, Australia, 2007. RMIT University.
- [JS01] H. L. Jones e M Snyder. Supervisory control of multiple robots based on a real-time strategy game interaction paradigm. 2001.
- [KPS⁺06] David B. Kaber, Carlene M. Perry, Noa Segall, Christopher K. McClernon e Lawrence J. Prinzel III. Situation awareness implications of adaptive automation for information processing in an air traffic control-related task. *International Journal of Industrial Ergonomics*, 36(5):447 – 462, 2006. Cognitive Engineering Insights for Human Performance and Decision Making.
- [KWSN06] David B. Kaber, Melanie C. Wright e Mohamed A. Sheik-Nainar. Investigation of multi-modal interface features for adaptive automation of a human-robot system. *Int. J. Hum.-Comput. Stud.*, 64(6):527–540, 2006.
- [LRY⁺07] Woohun Lee, Hokyoung Ryu, Gyunghye Yang, Hyunjin Kim, Youngkuk Park e Seokwon Bang. Design guidelines for map-based human-robot interfaces: A colocated workspace perspective. *International Journal of Industrial Ergonomics*, 37(7):589 – 604, 2007.
- [MDKY09] Mark Micire, Jill L. Drury, Brenden Keyes e Holly A. Yanco. Multi-touch interaction for robot control. In *Proceedings of the 13th international conference on Intelligent user interfaces*, IUI '09, pages 425–428, New York, NY, USA, 2009. ACM.
- [MDM⁺09] Ricardo Martins, Paulo Sousa Dias, Eduardo R. B. Marques, José Pinto, João Borges Sousa e Fernando Lobo Pereira. Imc: A communication protocol for networked vehicles and sensors,. In *IEEE Oceans Europe'09*, 2009.
- [Miy01] Shinji Miyake. Multivariate workload evaluation combining physiological and subjective measures. *International Journal of Psychophysiology*, 40(3):233 – 238, 2001.
- [MWKS06] R. K. M. Sheh M. W. Kadous e C. Sammut. Controlling heterogeneous semi-autonomous rescue robot teams. In *In Proceedings IEEE Conference on Systems, Man and Cybernetics*, 2006.
- [NGT⁺] Amir M. Naghsh, Jeremi Gancet, Andry Tanoto, Jacques Penders, Chris R. Roast e Michel Ilzkovitz. Human robot interaction in guardians.
- [Nie94] Jakob Nielsen. Enhancing the explanatory power of usability heuristics. In *CHI '94: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 152–158, New York, NY, USA, 1994. ACM.
- [Org07] North Atlantic Treaty Organization. Standard interfaces of uav control system (ucs) for nato uav interoperability. *North Atlantic Treaty Organization Brussels*, 2007.

REFERÊNCIAS

- [OW04] Dan R. Olsen, Jr. e Stephen Bart Wood. Fan-out: measuring human control of multiple robots. In *CHI '04: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 231–238, New York, NY, USA, 2004. ACM.
- [Par99] Lynne E. Parker. Adaptive heterogeneous multi-robot teams. *Neurocomputing*, 28:75–92, 1999.
- [pc1] Pc/104 test-bed for software gps receiver (sgr) and software defined radio (sdr) applications.
- [pcc] Piccolo command center. http://www.cloudcaptech.com/piccolo_command_center.shtm (visitado a 26 Jan 2011).
- [PDG⁺06] José Pinto, Paulo Sousa Dias, Gil M. Gonçalves, Rui Gonçalves, Eduardo Marques, João Borges Sousa e Fernando Lobo Pereira. Neptus – a framework to support a mission life cycle. In *7th IFAC Conference on Manoeuvring and Control of Marine Craft*, Lisboa, 09/2006 2006.
- [pic] Piccolo ii. http://www.cloudcaptech.com/piccolo_II.shtm/ (visitado a 26 Jan 2011).
- [PJS⁺10] Matthew S. Prewett, Ryan C. Johnson, Kristin N. Saboe, Linda R. Elliott e Michael D. Covert. Managing workload in human-robot interaction: A review of empirical studies. *Comput. Hum. Behav.*, 26(5):840–856, 2010.
- [Pow06] G. Bruch M. Powell, D. Gilbreath. Multi-robot operator control unit. In *Proceedings of 2006 SPIE THE INTERNATIONAL SOCIETY FOR OPTICAL ENGINEERING*, pages 6230–67, 2006.
- [SAF⁺00] Reid Simmons, David Apfelbaum, Dieter Fox, Robert P. Goldman, Karen Zita Haigh, David J. Musliner, Michael Pelican e Sebastian Thrun. Coordinated deployment of multiple, heterogeneous robots. In *In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2254–2260, 2000.
- [SFK⁺06] Aaron Steinfeld, Terrence Fong, David Kaber, Michael Lewis, Jean Scholtz, Alan Schultz e Michael Goodrich. Common metrics for human-robot interaction. In *HRI '06: Proceedings of the 1st ACM SIGCHI/SIGART conference on Human-robot interaction*, pages 33–40, New York, NY, USA, 2006. ACM.
- [SHB⁺04] Neville Stanton, Alan Hedge, Karel Brookhuis, Eduardo Salas e Hal Hendrick. *Handbook of human factors and ergonomics methods*. Chemical Rubber Company Press, 2004.
- [SSW⁺09] Paul M. Salmon, Neville A. Stanton, Guy H. Walker, Daniel Jenkins, Darshna Ladva, Laura Rafferty e Mark Young. Measuring situation awareness in complex systems: Comparison of measures study. *International Journal of Industrial Ergonomics*, 39(3):490 – 500, 2009. Selected papers from ECCE 2007, the 25th Anniversary Conference of the European Conference on Cognitive Ergonomics.

REFERÊNCIAS

- [WGG10] Jonathan M. Whetten, Michael A. Goodrich e Yisong Guo. Beyond robot fan-out: Towards multi-operator supervisory control. In *SMC*, pages 2008–2015, 2010.
- [WL08] Jijun Wang e Michael Lewis. Assessing cooperation in human control of heterogeneous robots. In *HRI '08: Proceedings of the 3rd ACM/IEEE international conference on Human robot interaction*, pages 9–16, New York, NY, USA, 2008. ACM.
- [Yan01] Holly A. Yanco. Designing metrics for comparing the performance of robotic systems in robot competitions. In *Proceedings of the Workshop on Measuring Performance and Intelligence of Intelligent Systems (PERMIS)*, 2001.
- [YDS04] Holly A. Yanco, Jill L. Drury e Jean Scholtz. Beyond usability evaluation: analysis of human-robot interaction at a major robotics competition. *Hum.-Comput. Interact.*, 19(1):117–149, 2004.

Anexo A

Simulações de Teste

Nesta secção estão documentados, na íntegra, os detalhes operacionais das simulações efectuadas ao módulo de operação.

A.1 Simulação 01

A simulação 01 foi efectuada a 7 de Janeiro de 2011, dentro do LSTS, de forma a testar a fiabilidade do módulo de operação numa situação de três UAV's activos com um supervisor. A área simulada foi a da OTA, onde foram delineadas zonas de operação para a equipa de UAV's, como é visível na Figura [A.1](#) . Seguidamente, são apresentados os detalhes da missão.

A.1.1 Constituição da Equipa

A equipa de trabalho foi constituída por cinco elementos:

- António Sérgio Ferreira - **Examinador**;
- Carlos Eduardo Oliveira - **TACO**;
- Filipe Melo Ferreira - **Op3**;
- Ricardo Ayres Bencatel - **Op2**;
- Rui Sérgio Caldeira - **Op1**.

Sendo que a cada um deles foi alocada uma função específica.

A.1.2 Objectivos

- Coordenação de três UAVs;
- Teste de interacção entre OPs e TACO;
- Condições operacionais:
 - Pista: BA2 (Ota);
 - Zona de aterragem/descolagem única - Pistas 17/35, junto à GS;

- Vento de Norte.
- Tarefas a operacionais:
 - UAV 1 – Vigilância de perímetro da GS;
 - UAV 2 – Vigilância de perímetro da Esquadra do PITVANT;
 - UAV 3 – Observação de alvos:
 - * Abrigo dos bombeiros;
 - * Topo sul da pista.

A.1.3 Plano de Operação

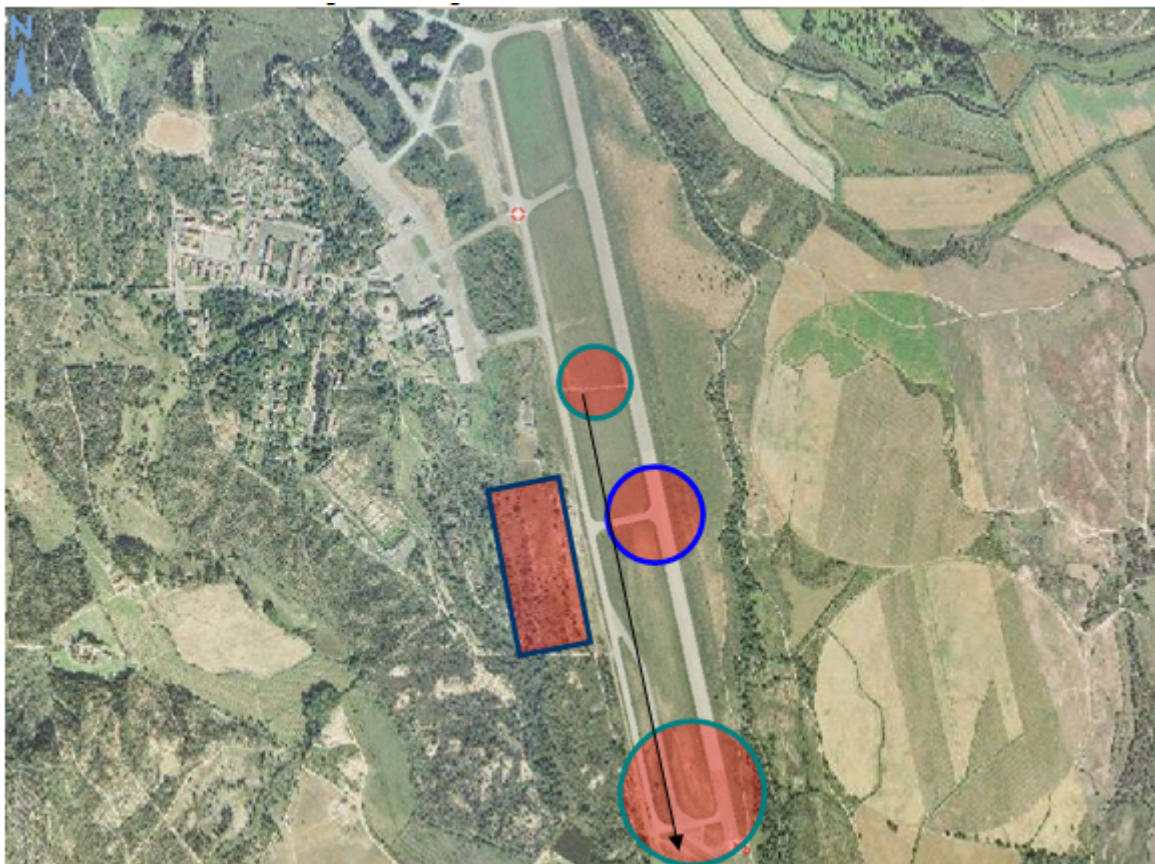


Figura A.1: Representação das áreas de vigilância do Simulacro 01

A.1.3.1 UAV 1 – Vigilância de perímetro da GS

- Trajecto:
 - Descolagem;
 - *Loiter* de 130m sobre a GS a 100m AGL;

- Durante descolagens e aterragens de outros UAVs: Ajuste de *loiter* para Sul ou Norte, respectivamente, e subida para 300m AGL;
 - Aterragem.
- Duração: Período de utilização da GS;
- Tolerância: Ajuste de trajecto para aterragem ou descolagem de outros UAVs.

A.1.3.2 UAV 2 – Vigilância de perímetro da Esquadra do PITVANT

- Trajecto:
 - Descolagem;
 - *Loiter* de 130m a Norte da GS para verificação do sistema de vídeo a 150m AGL;
 - Trajecto rectangular a 250m AGL a Oeste da *taxiway* principal, com sobrevoo da Esquadra do PITVANT na perna a Norte;
 - *Loiter* de espera para aterragem a 250m AGL, 100 m a Sul da GS entre a Pista 35 e a *taxiway* principal;
 - Aterragem.
- Duração: 30 minutos.

A.1.3.3 UAV 3 – Observação de alvo no abrigo dos bombeiros

- Trajecto:
 - Descolagem;
 - *Loiter* de 130m a Norte da GS para verificação do sistema de vídeo a 150m AGL;
 - *Loiter* de 100m sobre o abrigo dos bombeiros (a Norte da GS) a 200m AGL;
 - Transição para topo Sul da Pista 32 a 200m AGL;
 - *Loiter* de 200m sobre o topo Sul da Pista 32 a 200m AGL;
 - *Loiter* de espera para aterragem a 200m AGL, 100 m a Sul da GS e a Este da Pista 35;
 - Aterragem.
- Duração: 25 minutos.

A.1.4 Análise NASA-TLX

Segue-se a análise de *workload*, utilizando o método NASA-TLX. São apresentados, pela seguinte ordem, os pesos atribuídos a cada componente (Tabela A.1); os valores de *workload* não pesados para o módulo antigo (Tabela A.2) e para o módulo novo (Tabela A.3); finalmente, são apresentados os valores de *workload* pesados para o módulo antigo (Tabela A.4) e para o módulo novo (Tabela A.5).

A.1.4.1 Peso das Componentes

Tabela A.1: Tabela de distribuição de pesos entre componentes de *workload*

Weight Distribution (0-5)	
Mental Demand	4
Physical Demand	0
Temporal Demand	4
Performance	2
Effort	4
Frustration	1

A.1.4.2 Workload Não Pesado das Componentes

Tabela A.2: Tabela de distribuição do *workload*, não pesado, entre componentes, no módulo antigo

Unweighted Workload Distribution (0-100)	
Mental Demand	74
Physical Demand	55
Temporal Demand	84
Performance	31
Effort	89
Frustration	79

Tabela A.3: Tabela de distribuição do *workload*, não pesado, entre componentes, no módulo novo

Unweighted Workload Distribution (0-100)	
Mental Demand	39
Physical Demand	34
Temporal Demand	39
Performance	11
Effort	44
Frustration	10

A.1.4.3 Workload Pesado das Componentes

Tabela A.4: Tabela de distribuição do *workload*, pesado, entre componentes, no módulo antigo

Weighted Workload Distribution (%)	
Mental Demand	19,24
Physical Demand	0
Temporal Demand	22,68
Performance	4,03
Effort	24,03
Frustration	5,53

Tabela A.5: Tabela de distribuição do *workload* pesado, entre componentes, no módulo novo

Weighted Workload Distribution (%)	
Mental Demand	10,14
Physical Demand	0
Temporal Demand	10,53
Performance	1,43
Effort	11,88
Frustration	0,7

A.1.5 Análise SAGAT

Nesta secção, é apresentada a análise da SA do supervisor, utilizando o método SAGAT. São apresentadas as distribuições de SA, ao longo de dez interrupções, no módulo antigo (Tabela A.6) e no módulo novo (Tabela A.7). Os detalhes relativos às questões efectuadas podem ser consultados no Anexo C.

Tabela A.6: Tabela de distribuição da SA no módulo antigo

Weighted Workload Distribution (%)						
	Stop 1	Stop 2	Stop 3	Stop 4	Stop 5	
Query 1	3/3	2/3	3/3	1/3	2/3	
Query 2	1	0	1	0	1	
Query 3	0	0	0	0	1	
Query 4	1	1	1	0	1	
Query 5	1	1	1	0	1	
Query 6	1	0	0	0	1	
Query 7	1	1	1	1	0	
	Stop 6	Stop 7	Stop 8	Stop 9	Stop 10	Total
Query 1	3/3	1/3	2/3	3/3	1/3	0,70
Query 2	0	1	1	0	1	0,60
Query 3	0	1	1	0	0	0,30
Query 4	1	1	0	0	1	0,70
Query 5	0	1	1	1	1	0,80
Query 6	1	0	1	1	1	0,60
Query 7	0	1	1	0	1	0,70

Tabela A.7: Tabela de distribuição da SA no módulo novo

Weighted Workload Distribution (%)						
	Stop 1	Stop 2	Stop 3	Stop 4	Stop 5	
Query 1	3/3	2/3	3/3	2/3	3/3	
Query 2	1	1	1	1	1	
Query 3	1	1	1	0	1	
Query 4	1	0	1	1	1	
Query 5	1	0	1	1	0	
Query 6	1	0	1	0	1	
Query 7	1	1	1	1	1	
	Stop 6	Stop 7	Stop 8	Stop 9	Stop 10	Total
Query 1	3/3	3/3	2/3	3/3	3/3	0,90
Query 2	1	1	0	0	1	0,80
Query 3	0	1	1	1	0	0,70
Query 4	0	1	0	1	0	0,60
Query 5	0	1	1	1	1	0,70
Query 6	1	0	1	1	1	0,70
Query 7	0	1	1	0	1	0,80

A.1.6 Questionário de Avaliação

Finalmente, é apresentado o resultado do questionário, efectuado no final do simulacro, que fornece uma avaliação dos componentes do módulo novo efectuada pelo supervisor.

A.1.6.1 Questões

1. Considere os componentes do novo módulo de operações. Indique, por ordem decrescente, a importância que esses componentes tiveram para o sucesso da sua tarefa.
2. Justifique a importância dada ao primeiro componente da listagem anterior.
3. Justifique a importância dada ao último componente da listagem anterior.
4. Sente que os novos ícons dos veículos são úteis?
 - (a) O que contribuiu para essa conclusão?
5. Sente que os novos ícons para as fases de voo são úteis?
 - (a) O que contribuiu para essa conclusão?
6. Qual sente que foi a maior mudança deste módulo para o anterior?
7. Que melhorias faria ao novo módulo?
8. Está satisfeito com o novo módulo?

A.1.6.2 Respostas

1.
 - Uav Altitude Panel
 - Uav Supervisor Panel
 - Uav Status Panel
 - Generic Displays
 - System List
 - Uav Function Panel
 - Uav Mini Map Panel
 - Image Evaluation Panel
2. De todos os componentes visíveis, este era o que eu passava mais tempo a verificar. Durante a simulação, acabei por dividir o meu tempo entre o painel de supervisão geral e o das altitudes.
3. Relativamente a este, acabou por ser um empate. Em relação ao painel de avaliação de imagens não foi possível avaliá-lo, portanto não posso ter uma boa ideia acerca dele. Em relação ao mini mapa, devido ao espaço da missão ser tão pequeno, não senti necessidade em usá-lo. Talvez em ambientes maiores ganhe esse hábito.
4. Sim, bastante.
 - (a) O facto de ser possível ter uma noção rápida de quais UAV's precisam da minha atenção alivia-me bastante o trabalho.
5. Sim.
 - (a) Num cenário com mais aviões, e especialmente em situações iniciais e finais de missão, é sempre interessante ter essa informação de forma rápida.
6. A adição de um painel que permita ver, em simultâneo, todas as altitudes actuais da equipa de UAV's é uma dádiva. Até agora, tínhamos de recorrer ao *renderer* 3D para verificar as altitudes, o que se tornava complicado. Isto permite aliviar muitas situações de stress que surgem durante o decorrer de uma missão.
7. Adicionaria uma opção para poder seleccionar os aviões através do painel de altitude, visto ser tão usado. Era também uma boa ideia permitir saber, no painel principal, a que operador está associado cada avião. Finalmente, era interessante se a mudança de estado de voo fosse feita de forma automática.
8. Sim, bastante.

A.2 Simulação 02

A simulação 02 foi efectuada a 14 de Janeiro de 2011, dentro do LSTS, de forma a testar a fiabilidade do módulo de operação numa situação de dois UAV's activos com um supervisor. A área simulada foi a da OTA, onde foram delineadas zonas de operação para a equipa de UAV's, como é visível na Figura A.2 . Seguidamente, são apresentados os detalhes da missão.

A.2.1 Constituição da Equipa

A equipa de trabalho foi constituída por quatro elementos:

- António Sérgio Ferreira - **Examinador**;
- Filipe Melo Ferreira - **Op2**;
- Ricardo Haires Bencatel - **TACO**;
- Rui Sérgio Caldeira - **Op1**.

Sendo que a cada um deles foi alocada uma função específica.

A.2.2 Objectivos

- Coordenação de dois UAVs;
- Teste de interacção entre OPs e TACO;
- Condições operacionais:
 - Pista: BA2 (Ota);
 - Zona de aterragem/descolagem única - Pistas 17/35, junto à GS;
 - Vento de Norte.
- Tarefas a operacionais:
 - UAV 1 – Vigilância de perímetro da GS;
 - UAV 2 – Vigilância de perímetro da Esquadra do PITVANT.

A.2.3 Plano de Operação

A.2.3.1 UAV 1 – Vigilância de perímetro da GS

- Trajecto:
 - Decolagem;
 - *Loiter* de 150m sobre a GS a 100m AGL;
 - Durante descolagens e aterragens de outros UAVs: Ajuste de loiter para Sul ou Norte, respectivamente, e subida para 250m AGL;
 - Aterragem.
- Duração: período de utilização da GS;
- Tolerância: Ajuste de trajecto para aterragem ou descolagem de outros UAVs.



Figura A.2: Representação das áreas de vigilância do Simulacro 02

A.2.3.2 UAV 2 – Vigilância de perímetro da Esquadra do PITVANT

- Trajecto:
 - Descolagem;
 - *Loiter* de 130m a Norte da GS para verificação do sistema de vídeo a 150m AGL;
 - Trajecto rectangular a 250m AGL a Oeste da *taxiway* principal, com sobrevoo da Esquadra do PITVANT na perna a Norte;
 - *Loiter* de espera para aterragem a 250m AGL, 100 m a Sul da GS entre a Pista 35 e a *taxiway* principal;
 - Aterragem.
- Duração: 30 minutos.

A.2.4 Análise NASA-TLX

Segue-se a análise de *workload*, utilizando o método NASA-TLX. São apresentados, pela seguinte ordem, os pesos atribuídos a cada componente (Tabela A.8); os valores de *workload* não pesados para o módulo antigo (Tabela A.9) e para o módulo novo (Tabela

A.10); finalmente, são apresentados os valores de *workload* pesados para o módulo antigo (Tabela A.11) e para o módulo novo (Tabela A.12).

A.2.4.1 Peso das Componentes

Tabela A.8: Tabela de distribuição de pesos entre componentes de *workload*

Weight Distribution (0-5)	
Mental Demand	4
Physical Demand	0
Temporal Demand	4
Performance	2
Effort	3
Frustration	2

A.2.4.2 Workload Não Pesado das Componentes

Tabela A.9: Tabela de distribuição do *workload*, não pesado, entre componentes, no módulo antigo

Unweighted Workload Distribution (0-100)	
Mental Demand	62
Physical Demand	44
Temporal Demand	66
Performance	22
Effort	67
Frustration	53

Tabela A.10: Tabela de distribuição do *workload*, não pesado, entre componentes, no módulo novo

Unweighted Workload Distribution (0-100)	
Mental Demand	39
Physical Demand	36
Temporal Demand	42
Performance	12
Effort	43
Frustration	13

A.2.4.3 Workload Pesado das Componentes

Tabela A.11: Tabela de distribuição do *workload*, pesado, entre componentes, no módulo antigo

Weighted Workload Distribution (%)	
Mental Demand	16,74
Physical Demand	0
Temporal Demand	17,82
Performance	2,86
Effort	13,40
Frustration	6,89

Tabela A.12: Tabela de distribuição do *workload*, pesado, entre componentes, no módulo novo

Weighted Workload Distribution (%)	
Mental Demand	10,53
Physical Demand	0
Temporal Demand	11,34
Performance	1,56
Effort	8,60
Frustration	1,69

A.2.5 Análise SAGAT

Nesta secção, é apresentada a análise à SA do supervisor, utilizando o método SAGAT. São apresentadas as distribuições de SA, ao longo de dez interrupções, no módulo antigo (Tabela A.13) e no módulo novo (Tabela A.14). Os detalhes relativos às questões efectuadas podem ser consultados no Anexo C.

Tabela A.13: Tabela de distribuição da SA no módulo antigo

Weighted Workload Distribution (%)						
	Stop 1	Stop 2	Stop 3	Stop 4	Stop 5	
Query 1	2/2	2/2	2/2	1/2	2/2	
Query 2	1	1	1	0	1	
Query 3	1	0	1	0	1	
Query 4	1	1	1	1	1	
Query 5	1	1	1	0	1	
Query 6	1	1	0	0	1	
Query 7	1	0	1	1	0	
	Stop 6	Stop 7	Stop 8	Stop 9	Stop 10	Total
Query 1	2/2	2/2	2/2	1/2	2/2	0,90
Query 2	0	1	1	0	1	0,70
Query 3	0	1	1	0	1	0,60
Query 4	1	1	1	0	1	0,90
Query 5	1	1	1	0	1	0,80
Query 6	1	0	1	1	1	0,70
Query 7	1	1	1	0	1	0,70

Tabela A.14: Tabela de distribuição da SA no módulo novo

Weighted Workload Distribution (%)						
	Stop 1	Stop 2	Stop 3	Stop 4	Stop 5	
Query 1	2/2	2/2	2/2	2/2	2/2	
Query 2	1	1	1	1	0	
Query 3	0	1	1	1	1	
Query 4	0	1	1	1	1	
Query 5	1	0	1	1	1	
Query 6	1	1	0	0	1	
Query 7	1	1	1	1	1	
	Stop 6	Stop 7	Stop 8	Stop 9	Stop 10	Total
Query 1	2/2	2/2	2/2	2/2	2/2	1,00
Query 2	1	1	1	1	1	0,90
Query 3	0	0	1	1	1	0,70
Query 4	0	1	1	1	1	0,80
Query 5	1	1	1	1	0	0,80
Query 6	1	0	1	1	1	0,70
Query 7	1	1	0	0	1	0,80

A.2.6 Questionário de Avaliação

Finalmente, é apresentado o resultado do questionário, efectuado no final do simulacro, que fornece uma avaliação dos componentes do módulo novo efectuada pelo supervisor.

A.2.6.1 Questões

1. Considere os componentes da novo módulo de operações. Indique, por ordem decrescente, a importância que esses componentes tiveram para o sucesso da sua tarefa.
2. Justifique a importância dada ao primeiro componente da listagem anterior.
3. Justifique a importância dada ao último componente da listagem anterior.
4. Sente que os novos ícons dos veículos são úteis?
 - (a) O que contribuiu para essa conclusão?
5. Sente que os novos ícons para as fases de voo são úteis?
 - (a) O que contribuiu para essa conclusão?
6. Qual sente que foi a maior mudança deste módulo para o anterior?
7. Que melhorias faria ao novo módulo?
8. Esta satisfeito com o novo módulo?

A.2.6.2 Respostas

1.
 - Uav Altitude Panel
 - Uav Supervisor Panel
 - Uav Status Panel
 - Generic Displays
 - System List
 - Image Evaluation Panel
 - Uav Function Panel
 - Uav Mini Map Panel
2. De todos os componentes à minha disposição o *Altitude Panel* era ao que eu recorria mais. Recordo agora missões onde este painel teria sido muito útil.
3. Relativamente ao mini mapa, devido ao espaço da missão ser tão pequeno, não senti que este fosse muito útil. É mais fácil, por enquanto, manipular só o painel principal.
4. Sim, bastante.
 - (a) Não só é muito útil conseguir ter informação acerca do estado do UAV só de olhar para ele, mas a própria forma como ele está concebido aumenta o contraste com o mapa. Assim consigo ter uma melhor noção de onde o UAV está no meio do plano de missão.
5. Sim.
 - (a) Tendo sido supervisor várias vezes, para mim é sempre útil ter uma noção rápida da fase de voo em que se encontram os UAV's da missão.
6. Definitivamente o painel que nos permite ver todas as altitude da equipa em simultâneo. Agora não é preciso fazer mais malabarismos para conseguir ter essa informação, o que torna as missões muito mais tranquilas.
7. A maior melhoria que faria era tornar a mudança de estado de voo de forma automática.
8. Sim.

Anexo B

Ferramenta NASA-TLX

Nesta secção, é possível encontrar expostos os pormenores dos questionários NASA-TLX utilizados para determinar o *workload* verificado em cada simulação efectuada.

B.1 Determinação do Peso da Componente

Para determinar o peso de cada componente de avaliação no *workload* total da tarefa, é utilizado um método de escolha entre pares de componentes. Assim sendo, são apresentados, ao elemento avaliado, quinze pares (Figura B.1).

Effort or Performance	Temporal Demand or Frustration	Frustration or Effort	Performance or Mental Demand
Temporal Demand or Effort	Physical Demand or Frustration	Performance or Temporal Demand	Mental Demand or Effort
Performance or Frustration	Physical Demand or Temporal Demand	Mental Demand or Physical Demand	Effort or Physical Demand
Physical Demand or Performance	Temporal Demand or Mental Demand	Frustration or Mental Demand	

Figura B.1: Exemplo dos pares de avaliação de pesos para cada componente do *workload*

À medida que estes são mostrados, um a um, o elemento avaliado tem de identificar qual dos dois contribui mais para o *workload* da tarefa.

B.2 Determinação dos Valores Relativos

Após recolher os pesos para cada componente, é utilizada uma escala, presente na Figura B.2, onde o elemento avaliado marca o *workload* sentido para cada componente.

Mental Demand	How mentally demanding was the task?
Very Low	Very High
Physical Demand	How physically demanding was the task?
Very Low	Very High
Temporal Demand	How hurried or rushed was the pace of the task?
Very Low	Very High
Performance	How successful were you in accomplishing what you were asked to do?
Perfect	Failure
Effort	How hard did you have to work to accomplish your level of performance?
Very Low	Very High
Frustration	How insecure, discouraged, irritated, stressed, and annoyed were you?
Very Low	Very High

Figura B.2: Exemplo da escala usada para recolher os valores de *workload* para cada componente

A performance funciona contrariamente ao *workload*; ou seja, quanto maior for a performance sentida pelo elemento avaliado, menor será o *workload* sentido por ele. Daí a escala "crescer" da direita para a esquerda.

B.3 Determinação dos Valores Pesados

Recolhidos os dados iniciais, torna-se então necessário fazer os cálculos ajustados relativamente aos pesos, seguindo a fórmula:

$$ComponentRating : \quad R = \frac{ComponentWeight}{TotalPairs} \times (RelativeWeight) \quad (B.1)$$

Anexo C

Ferramenta SAGAT

Nesta secção, é possível encontrar expostos os pormenores dos questionários SAGAT utilizados para determinar a SA verificada em cada simulação efectuada.

C.1 Modelo

De forma a averiguar a SA do supervisor durante os simulacros, optou-se pelo seguinte grupo de questões:

- Point-out the position of each UAV currently active;
- Single-out the currently selected UAV;
- Determine the altitude of all active UAV's;
- Determine the altitude of the selected UAV;
- Determine the speed of the selected UAV;
- Determine the current UAV's waypoint;
- Determine the current UAV's ID.

C.2 Metodologia

Seguindo as recomendações presentes em [SHB⁺04], antes de iniciar os questionários, foi dado um período de cinco minutos de adaptação ao supervisor. Seguidamente, foram efectuadas as questões acima delineadas, sem nenhuma ordem pré-definida, a cada cinco minutos. Visto ser impossível pausar a simulação, de forma a questionar o supervisor, foi tapado o seu monitor a cada interrupção de forma que este não conseguisse aceder à informação perguntada.

Anexo D

Manual de Inicialização do Módulo

Nesta secção, serão fornecidas as informações necessárias para utilizar o módulo de controlo tendo em consideração os processos de inicialização do Neptus, PCC e Dune. A ordem de exposição seguida por este manual é a ordem pela qual devem ser inicializados cada um dos três componentes essenciais à utilização do módulo.

D.1 Preparação Inicial

Todas as aplicações podem ser corridas no mesmo PC, não sendo, no entanto, aconselhável, devido ao consumo elevado de recursos que iria pôr em causa o aproveitamento e qualidade da missão. Assim sendo, o aconselhado será o uso de PC's individuais para cada elemento da equipa de UAV's. Em uma das máquinas responsáveis por simular um UAV, devem constar os softwares Dune e PCC. O supervisor apenas necessita de possuir o Neptus na sua máquina.

É necessário a existência de uma máquina virtual em cada um dos PC's dos operadores (consideram-se operadores os elementos da equipa que simulam os UAV's) de forma a ser possível utilizar os *softwares* Dune e PCC. A utilização do Dune é aconselhável em ambiente Linux, devido à facilidade em pormenores de configuração, enquanto que a utilização do PCC necessita ser feita em ambiente Windows.

D.2 Piccolo Command Center

A primeira aplicação a ser iniciada é o PCC, bastando primeiro verificar os pormenores de comunicação, visível na Figura D.1. Isto é necessário para garantir que a emissão de mensagens esteja a correr dentro da mesma rede.

De seguida, é apenas necessário recorrer ao ícone do Piccolo Command Center para iniciar a aplicação. Este irá automaticamente reconhecer UAV's activos na rede e fará a sua representação e a dos seus planos de voo. Como é possível verificar na Figura D.2, o UAV inicia a missão no solo, sendo o seu comando inicial o de *Pre-Launch*. É nesta fase que são efectuadas as verificações ao estado dos vários componentes do UAV e das

Manual de Inicialização do Módulo

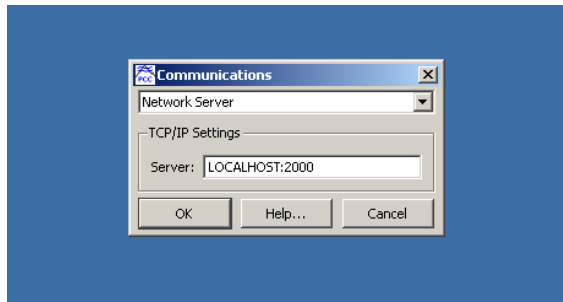


Figura D.1: Verificação da comunicação PCC

condições de missão, através de uma série de *checklists* protocolares.

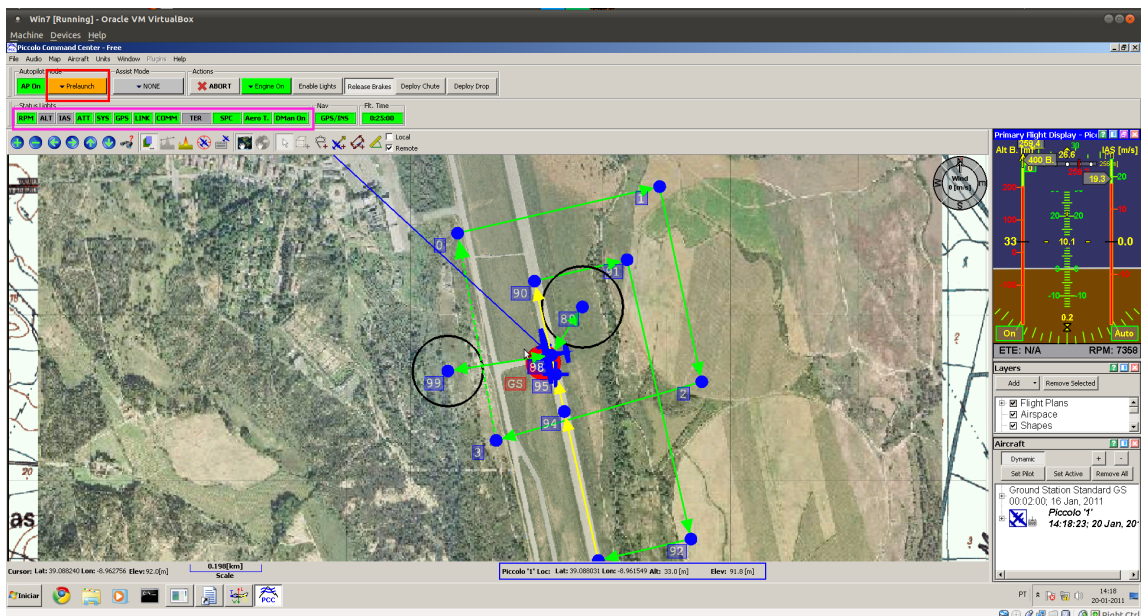


Figura D.2: Interface inicial do PCC

Dando início ao voo do UAV, é então possível que o seu operador verifique o evoluir da missão (Figura D.3), bem como alterar os planos do UAV durante o seu voo, caso seja necessário.

Terminam assim os passos necessários para a inicialização do PCC numa situação de missão com UAV's.

D.3 Dune

Para dar início ao Dune, é apenas necessário, dentro do ambiente Linux, aceder ao directório onde reside a *build* mais recente do software e, através do uso de uma consola

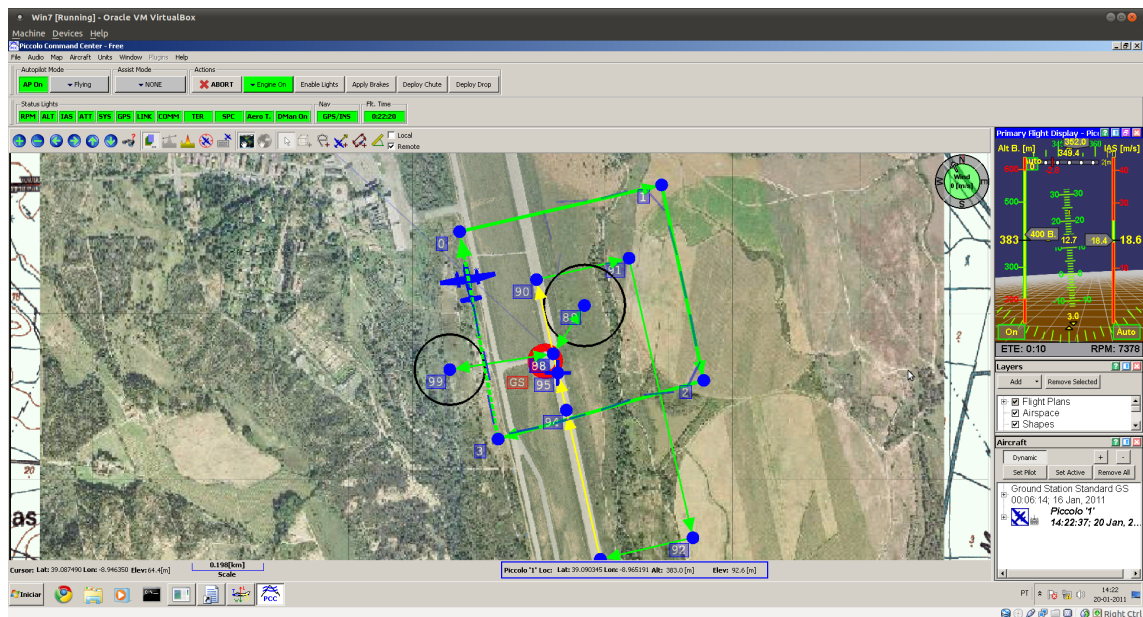


Figura D.3: Exemplo da interface do módulo Neptus

de comando, executar a seguinte instrução:

- `sudo ./dune -c uav-pcc.`

Após isto, é iniciado o *broadcast* de mensagens entre o UAV e o Dune, sendo possível verificar o pormenores de mensagem através do *prints* a ocorrer na consola de comando.

D.4 Neptus

Finalmente, torna-se possível iniciar o Neptus, sendo apresentado o ecrã inicial de opções (Figura D.4). O Neptus é a última componente a ser activada pois, na situação actual do LSTS, em missões UAV, o Neptus actua apenas como *listener* do Dune, conseguindo somente permissão para visualizar o que ocorre numa missão aérea.

Escolhida a consola de supervisor (Figura D.5), é então possível iniciar as funcionalidades de supervisor de missão. Caso não haja ainda nenhum UAV a emitir mensagens a interface da consola aparecerá vazia, como no exemplo da Figura D.6.

Conclui-se assim o manual de inicialização do módulo de operações.

Manual de Inicialização do Módulo

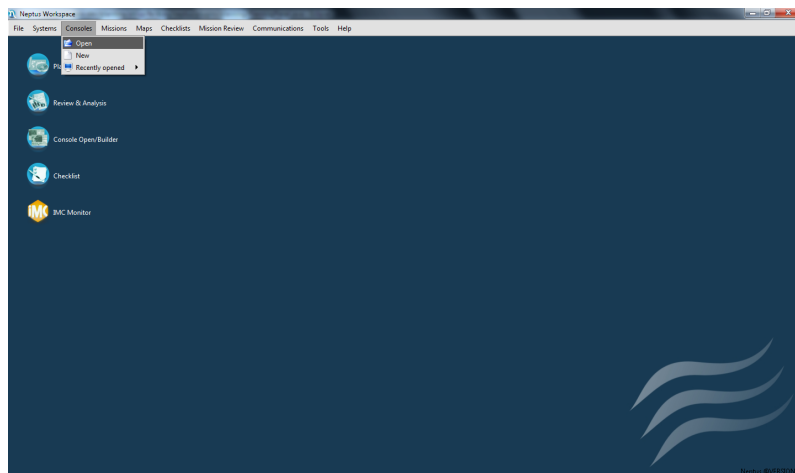


Figura D.4: Interface inicial do Neptus

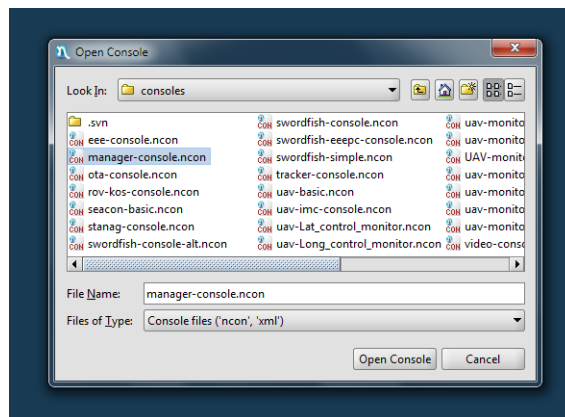


Figura D.5: Exemplo da interface do módulo Neptus

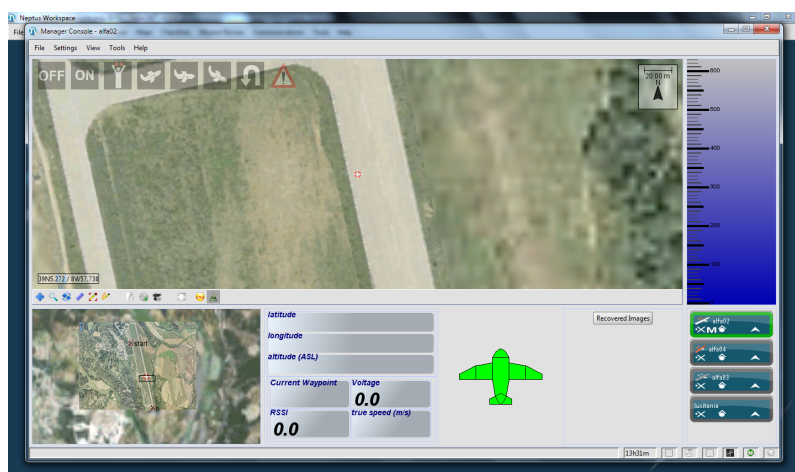


Figura D.6: Exemplo da interface do módulo Neptus