Projeto de software para simulação de vôo empregando a ferramenta Analytice II

Gustavo Augusto Domeneghine Tiveroli [1]
Paulo Cézar Stadsisz [1]
Jean Marcelo Simão [1 2]

Resumo: Este artigo apresenta a utilização da ferramenta de simulação Analytice II no desenvolvimento de um ambiente computacional de simulação de voo tridimensional. O processo de desenvolvimento envolve três fases de modelagem: geométrica, cinemática e comportamental, que incluem, respectivamente, a criação de um modelo 3D de uma aeronave, a definição de seus ângulos e limites de movimento e, por fim, a programação por meio da linguagem C++. A modelagem comportamental é a que determina as funções e o comportamento da aeronave no espaço tridimensional do animador gráfico interno do simulador. Testes com esse ambiente de simulação foram realizados com o modelo de um avião monomotor, programado para ligar a hélice, decolar e realizar rolagens.

Palavras-chave: Simulação computacional. Simulação de voo. Analytice II.

Abstract: *This article presents the usage of the simulation tool called Analytice II in the development of a tridimensional flight simulation computational environment. The development process involves three modelling steps: geometric, cinematic and behavioral, that includes respectively, the creation of an aircraft 3D model; the definition of its movement angles and limits; and finally the programming using the C++ language. The behavioral model determines the functions and the behavior of the aircraft in the tridimensional space of the internal graphical animator of the simulation tool. Tests are being conducted on the model of a beechcraft, programmed to turn on the propeller, take off and perform rolls.*

Keywords: *Computational simulation. Flight simulation. Analytice II.*

1 Introdução

Simular é uma forma de representar o funcionamento de um sistema ou processo real copiando suas características e imitando suas operações no mundo real, seja por meio de sua execução manual, seja computacional. O objetivo geral da simulação é analisar o comportamento desse sistema de forma abstrata, a fim de obter resultados sem a necessidade de se dispor do sistema real. A inteção é diminuir custos e riscos de engenharia e contribuir para o aumento no desempenho e qualidade do produto, permitindo que o usuário teste várias soluções de projetos sem comprometer recursos.

A observação do sistema, realizada por meio da simulação, levará a melhores resultados se houver um alto grau de conhecimento sobre o sistema e seu contexto, ou seja, quanto mais suas características forem conhecidas e incorporadas na simulação, maiores serão as chances de se obterem resultados mais confiáveis por meio da simulação. Particularmente, a simulação computacional envolve o desenvolvimento descritivo de modelos computacionais e possui a função de exercitar esses modelos para que sejam capazes de prever o desempenho e comportamento operacional do sistema a ser modelado.

[1]Programa de Pós-Graduação em Engenharia Elétrica e Informática Industrial (CPGEI) - Universidade Tecnológica Federal do Paraná (UTFPR), Campus Curitiba/Central - Av. Sete de Setembro, 3165- Rebouças - CEP 80230-901 - Curitiba (PR) - Brasil
{gustavoaugusto47@gmail.com, stadsisz@utfpr.edu.br, jeansimao@utfpr.edu.br}
[2]Programa de Pós-Graduação em Computação Aplicada (PPGCA/DAINF) - Universidade Tecnológica Federal do Paraná (UTFPR), Campus Curitiba/Central - Av. Sete de Setembro, 3165- Rebouças - CEP 80230-901 - Curitiba (PR) - Brasil

doi: 10.5335/rbca.2011.1772

Simulação é um método indispensável na resolução de diversos problemas existentes no mundo real. Devido a isso, o número de aplicações que utilizam simulação está crescendo, passando a ser incorporada diariamente por algumas companhias, pois os benefícios da simulação vão além de somente fornecer previsão do comportamento do sistema [1]. Para a pesquisa, a simulação é um recurso extremamente importante, porque envolve o desenvolvimento de conhecimento (operacional) à priori de uma variedade de tópicos incluindo um entendimento detalhado da situação a ser modelada, coleção de dados e análise, *design* e condução de experimentos e a interpretação dos resultados no contexto em que o problema foi modelado.

As mais diversas áreas do conhecimento fazem uso da simulação antes de aplicar no mundo real o que seria a possível solução para um problema. Gerar um modelo abstrato de determinado processo ou sistema real a fim de testar suas funcionalidades pode ser muito mais seguro, pois é possível antecipar acontecimentos e explorar os comportamentos do sistema e de todas suas variáveis e características. A relação custo/tempo apresentaria melhor retorno financeiro ao projeto a longo prazo uma vez que falhas podem ser identificadas e corrigidas no início, visando ganho em termos de desenvolvimento do sistema e segurança do projeto. O custo típico de um estudo de simulação é substancialmente menor que 1% da quantia total gasta para a implementação de um projeto, e considerando que o custo de uma alteração no sistema depois de sua instalação é grande demais, a simulação é um sábio investimento [1].

Um exemplo de domínio pertinente de aplicação de simulação é o aeronáutico. As simulações de voo, operadas por pilotos dentro ou fora (de estações de controle) das aeronaves, ou até mesmo executadas por um computador, são tarefas diárias das forças armadas e empresas de aviação civil. Simulações neste âmbito possuem diversas vantagens e contribuem para a evolução da tecnologia aeroespacial, na qual testes exaustivos são realizados confrontando os limites aerodinâmicos existentes. A diminuição de custos empregando-se simulações em ambos os contextos de avião tripulado ou não, é uma das principais vantagens [1], [2], [3]. A simulação é realizada sobre um modelo, o qual possui um conjunto de suposições que são relacionadas à operação do sistema.

O artigo trata de um simulador de voo, criado utilizando a ferramenta Analytice II com padrões de simulação e modelagem. Neste trabalho, o modelo da aeronave inclui o modelo geométrico e possui suas funcionalidades de movimentação e animação estabelecidas por meio de uma modelagem cinemática. Porém, para que esse modelo seja propriamente simulado, uma modelagem comportamental também é desenvolvida, utilizando uma ferramenta computacional e linguagem de programação na criação de algoritmos de comportamento da aeronave, de planos de voo e tarefas que a aeronave desempenha ao longo de sua missão.

2 Simulação computacional

A simulação computacional é uma tentativa de modelar elementos do mundo real ou situação hipotética em um computador, para que possam ser estudados para buscar mais conhecimento sobre como o sistema funciona. Pela mudança de variáveis, predições podem ser feitas sobre o comportamento do sistema ou, em outras palavras, desenvolver um modelo de um sistema real ou imaginário e conduzir experimentos com esse modelo [2].

Como ferramenta de modelagem para áreas como, por exemplo, física, química, biologia, economia, ciências sociais e engenharia, as simulações computacionais têm se tornado essenciais nessas áreas. Projetos nestas áreas eram, tradicionalmente, modelados por meio de modelos matemáticos, que tentavam encontrar soluções analíticas a partir de um conjunto de parâmetros e condições iniciais. Entretanto, os modelos analíticos, ao contrário da simulação computacional, não são capazes de enumerar todos os possíveis estados nem de gerar exemplos de cenários representativos [4].

Um exemplo onde sistemas analíticos não seriam suficientes seria a análise de voos. Neste contexto, a simulação computacional se faz pertinente. O emprego da simulação computacional na área de aplicação de simulação de voo em aeronaves não tripuladas possui algumas vantagens, como obter uma visão do ambiente em que a aeronave se encontra e realizar testes de aerodinâmica em situações adversas, além de analisar os parâmetros e o funcionamento da simulação. Ainda, uma simulação de voo pode envolver aspectos de simulação discreta e de simulação contínua.

Simulações podem ser definidas como Simulação a Eventos Discretos (SED) ou Simulação a Eventos Contínuos (SEC), determinados pela maneira como se alteram os estados das variáveis. Simulações a eventos discretos

alteram suas variáveis em tempo real em momentos determinados, ao passo que simulações contínuas as alteram seguidamente por meio de uma função em que o tempo é uma variável. Geralmente, as simulações utilizam ambos os tipos de eventos, porém um deles é predominante e controla a simulação [5]. A Figura 1, a seguir, apresenta uma classificação dos tipos de simulação [1].

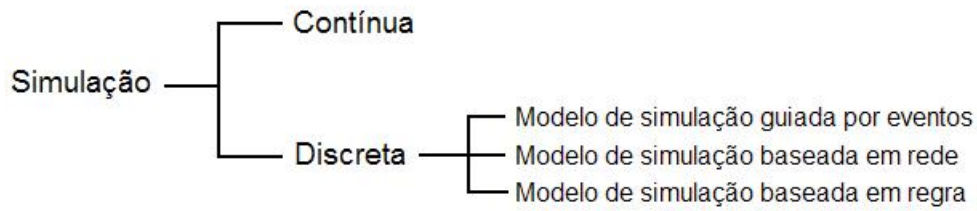

Figura 1. Relacionamento dos modelos de simulação

2.1 Simulação a eventos discretos

O diferencial da simulação a eventos discretos é sua habilidade de imitar a dinâmica do sistema real, responsável pela sua estrutura, sua função e sua maneira única de analisar os resultados. Por essa razão, este tipo de simulação é muito usado em sistemas como de manufatura e simulação de voo. No presente trabalho utiliza-se SED.

As definições básicas que descrevem a arquitetura de um sistema de simulação discreto são comuns para todos os sistemas de simulação [6] [7] [8]. Um resumo dessas definições é descrito a seguir:

- Relógio (clock): representa o tempo dentro da simulação. Por exemplo, em uma relação 1:n, 1 segundo simulado representa n segundos de tempo real. Em uma simulação com animação gráfica a atualização do modelo deve ser constante devido a mudança de estados. Em outros casos o relógio pode ser atualizado não baseado em constante.
- Evento: é um acontecimento responsável por causar mudanças de estado no sistema. Geralmente é acompanhado de temporizadores (time-stamps), que definem o momento da ocorrência de cada evento para que sejam gerados com antecedência e agendados em listas (a seguir). Um exemplo de evento em um sistema de simulação de voo é atingir determinada coordenada durante uma missão de exploração aérea de um local e capturar uma imagem atual daquele cenário.
- Estado: um conjunto de variáveis denominadas "variáveis de estado" define o estado do sistema. Em cada instante absoluto do sistema, o conjunto de valores dessas variáveis determina o respectivo estado do modelo.
- Mensagens: servem como notificação da ocorrência dos eventos. Podem também conter dados referentes à arquitetura do simulador.
- Tarefas ou processos: ambos possuem o mesmo significado e são responsáveis por modelar uma faceta do funcionamento do sistema real. Eventos podem ativá-los e, durante sua execução, podem alterar variáveis. Novos eventos também podem ser agendados.
- Listas: quase todos os simuladores a eventos discretos possuem listas como estrutura de dados para ordenar e armazenar atividades futuras. Alguns exemplos: lista contendo processos em funcionamento, como o controle de altitude, lista de eventos futuros (normalmente ordenados por time-stamps) e lista de processos que aguardam a liberação de um recurso para serem enviados para a lista de processos.
- Recursos: representam tanto os elementos abstratos como os elementos físicos do simulador. Um exemplo de elemento abstrato são as condições lógicas da simulação, e como elemento físico um exemplo é a própria aeronave ou parte dela, como sua hélice.

2.1.1 Estrutura da simulação a eventos discretos

Independentemente dos diversos paradigmas existentes nas simulações a eventos discretos, uma estrutura básica é utilizada na maioria dos programas e pacotes (*packages*) de simulação. Essa estrutura inclui os seguintes componentes: entidades, atividades e eventos, recursos, variáveis globais, um gerador de números aleatórios, um calendário, variáveis de estado do sistema e coletores estatísticos. Cada componente será descrito e exemplificado a seguir utilizando exemplos com simulação de voo.

- Entidades: as entidades são responsáveis pela mudança de estado do sistema e, consequentemente, pelo progresso da simulação a cada passo. Se nenhuma entidade estiver ativa no sistema, isso pode ser utilizado como recurso de parada da simulação. Entidades possuem atributos, que são as características únicas de cada entidade em particular e são necessários para sua função e desempenho na simulação. Um exemplo de entidade existente no modelo deste trabalho é a aeronave, que possui atributos que definem seu estado de movimento, chamados *SteadyAircraft* quando está parada, e *FlyingAircraft* quando está voando. Além de algo físico, entidades podem ser fluxos de informação que causam alguma mudança no sistema, como um pacote em uma rede de computadores, um alerta de e-mail e um pedido de um cliente. Essas entidades também possuem atributos, por exemplo: um pacote em uma rede de computadores possui atributos como tamanho do pacote, destino do pacote e *timeout* do pacote.
- Atividades e eventos: atividades representam a lógica e os processos na simulação. Eventos são condições ou acontecimentos que ocorrem em um determinado momento no tempo causando uma mudança no estado do sistema. Uma entidade interage com uma atividade que cria eventos. Há três tipos principais de atividades: delays, queues e logic. Um delay representa uma porção de tempo associada a uma atividade que determina um atraso específico no tempo simulado. Na simulação de voo, por exemplo, um delay ocorre quando a aeronave está em movimento na pista antes de levantar voo. Assim que uma entidade inicia um intervalo é o momento em que um evento ocorre; neste caso, quando a aeronave começa a levantar voo. Uma queue é uma estrutura que registra as entidades que aguardam ordenadamente a liberação de um recurso específico ou de uma condição do sistema para ocorrer. Não há prazos de espera nas queues. Neste caso, uma condição é aguardada. Já em uma atividade logic, um evento somente irá alterar o estado do sistema por meio da manipulação do estado das variáveis ou por meio de uma decisão lógica. No exemplo de uma simulação de voo, a lógica existe nas decisões de rolagem da aeronave, se ela deve ou não se inclinar para a direita ou para a esquerda.
- Recursos: recursos são meios empregados pelo sistema e, também, considerados na simulação pela influência que podem exercer no comportamento simulado do sistema. Exemplos comuns de recursos são trabalhadores, máquinas e nós em uma rede de computadores. No caso da simulação de voo, pode-se considerar a pista de pouso como um recurso. Ao longo da simulação é possível acompanhar os *feedbacks* dos recursos, como utilização e custos. Ao se tratar de pacotes de simulação, dependendo da área de aplicação, há vários recursos disponíveis para utilização em bibliotecas. Algumas ferramentas permitem criar e modelar recursos próprios da área de simulação pretendida. Diferentemente das entidades, que são o objeto de interesse da simulação, recursos são elementos coadjuvantes cujo comportamento só interessa dentro do limite de sua influência sobre as entidades simuladas. Assim, na concepção do modelo de simulação do sistema, devem-se identificar apenas os recursos fundamentais que integrarão o modelo.
- Variáveis globais: uma variável global pode ser utilizada pelo sistema a qualquer momento, geralmente com alguma finalidade que envolve todo o processo de simulação. O termo "global" indica exatamente o contexto da variável, ou seja, a variável armazena algum dado representativo de algum estado que é genérico a todo o sistema simulado ou ao ambiente da simulação. Um exemplo de variável global no modelo em estudo é o tempo total da simulação. Este tempo é armazenado em uma variável e atualizado regularmente. Várias atividades são sincronizadas e registros feitos baseando-se nesta mesma variável. Por essa razão, é considerada uma variável global. O incremento de tempo do simulador, ou passo da simulação, é outro exemplo de variável global. As simulações podem conter várias variáveis de estado, porém todas contêm a variável de estado relativa ao tempo atual da simulação.
- Gerador de números aleatórios: para determinar os valores de tudo que é aleatório em uma simulação usa-se o gerador de números aleatórios como entrada. Este gerador é, geralmente, representado por uma rotina específica, que é responsável por obter como saída um número entre 0 e 1 para as distribuições aleatórias. Para se calcular os intervalos no sistema, por exemplo, usa-se o gerador.

- Calendário: o calendário armazena a lista de eventos agendados que irão ocorrer ao longo da simulação; funciona como uma agenda na qual estão relacionados os eventos que devem se produzir. A base do processo de simulação consiste em avançar o tempo e consultar a agenda sobre os eventos que devem ser processados a cada tempo. Só existe um calendário em cada simulação e os eventos já realizados continuam armazenados. Produz-se, assim, um histórico da trajetória da simulação. O mecanismo de implementação e a lógica de operação do calendário variam de acordo com os propósitos e ferramenta de simulação empregada, porém o conceito de calendário é sempre presente.
- Coletores estatísticos: baseados nos atributos das entidades, os coletores estatísticos são responsáveis por armazenar estados (estados dos recursos, por exemplo), valores das variáveis ou estatísticas de desempenho do sistema. Três tipos de estatísticas são coletadas: counts, time-persistent e tallies. Um exemplo de count no modelo é o número de alvos identificados pela aeronave em sua missão. O coletor estatístico *time-persistent* mede o valor do peso pelo tempo em cada variável do sistema. As tallies têm a função de coletar observações a respeito da simulação, por exemplo, o tempo em que cada entidade permanece no sistema.

3 Descrição do ambiente de simulação Analytice II e seu uso como um simulador de voo

Nesta seção se descreve Analytice II, incluindo o seu ambiente de animação, e os processos de desenvolvimento e execução do simulador de voo a partir da adaptação de Analytice II.

3.1 Analytice II

O Analytice II é um programa com fins educacionais e de pesquisa para o desenvolvimento de *software* em aplicações industriais. O objetivo de sua utilização é criar um recurso alternativo para o *design* e simulação de simuladores de voo, além de ser altamente modular e escalável, possuindo uma explícita separação entre as entidades de planta e de controle emuladas por meio de uma rede de comunicação virtual. Por essa razão, o simulador Analytice II pode ser considerado um simulador realístico [13].

Como sendo uma ferramenta de simulação a eventos discretos o Analytice II foi desenvolvido na Universidade Tecnológica Federal do Paraná (UTFPR) voltado inicialmente para sistemas de manufatura, permitindo realizar experimentos para determinar parâmetros qualitativos e quantitativos por meio de seus principais módulos: modelo de equipamentos, ambiente físico, animação geométrica, núcleo e monitoração de simulação [9], [10], [11].

Os modelos de equipamentos que formam as primitivas de sua modelagem são representados por modelos geométricos, cinemáticos e comportamentais. O modelo geométrico pode ser importado de sistemas CAD. O modelo cinemático associa eixos de translação e rotação do modelo geométrico, possibilitando sua animação tridimensional. O modelo comportamental representa a evolução dos estados do modelo e apresenta uma interface que recebe sinais de comando, como "ligar hélice", e enviar sinais de resposta, como "pouso concluído com sucesso". Os modelos comportamentais são descritos por classes em linguagem de programação C++ cujas instâncias (objetos) são executadas e interagem com o núcleo do simulador durante a simulação [12].

Para a construção do projeto de simulador de voo serão utilizadas as bibliotecas do simulador Analytice II, nas quais serão instanciados os objetos relacionados às partes importantes do modelo CAD da aeronave para gerar movimento e animação. No projeto de simulação de voo não serão criados outros modelos (equipamentos) [9], porém a função de animação gráfica já existente no Analytice II [10] é utilizada na animação do modelo de aeronave. O recurso de animação gráfica aplicado a simulações computacionais proporciona um maior entendimento do sistema sob estudo.

3.1.1 Características do simulador

O Analytice II possui características importantes de grande impacto no projeto de simulador de voo.

- Arquitetura de simulação modular: permite, além da integração com outros softwares, a adição de novos módulos.

- Arquitetura de simulação escalável: pode-se evoluir um modelo simples para um modelo mais complexo.
- Monitoração do funcionamento dos modelos: é permitido monitorar o estado das variáveis internas de um modelo, que possui seu comportamento determinado pelos processos que executa, pelos sinais de troca realizados com o controle e pelas interações com outros modelos (equipamentos).
- Animação gráfica: conforme ocorre a execução dos processos, os modelos são animados e visíveis ao usuário. Como suporte existe o controle dos quadros gerados por segundo pela simulação, que não interferem com o relógio da simulação, não acontecendo, assim, atrasos no tempo que possam comprometer os resultados dos experimentos.
- Monitoração: inclui a monitoração do estado do sistema, que permite ao usuário selecionar as variáveis que deseja observar durante a simulação; a monitoração da coleta de informações pelo controle, que possibilita a execução de algoritmos por meio de uma *Application programming interface* (API), e também a monitoração de dados para análise, que seriam usadas para gerar arquivos de dados da simulação para serem importados por ferramentas de análise numérica.
- Controle da simulação: controle geral de atividades como a animação dos modelos, envio de mensagens e a definição da velocidade das simulações (retardada, tempo real ou acelerada).

3.2 Animação Gráfica

A utilização da animação em uma simulação proporciona ao usuário uma maior segurança a respeito dos resultados obtidos devido à capacidade de visualização do comportamento do sistema simulado, além de fornecer a dinâmica do funcionamento do sistema no ambiente em que foi aplicado. A confiança do usuário tanto na simulação quanto no simulador aumenta quando ele se torna capaz de ver todo o sistema por meio de uma animação gráfica [14], que é atualmente muito utilizada na simulação computacional [10], diferentemente das simulações que apresentam somente relatórios com as saídas do sistema.

A animação gráfica existente no simulador Analytice II envolve técnicas de implementação, modelagem da geometria e cinemática dos objetos e formatos de representação interna dos dados utilizando as bibliotecas OpenGL.

3.3 Uso do Analytice II como simulador de voo

Nesta seção apresentam-se as modelagens realizadas no processo de desenvolvimento do simulador de voo usando as modelagens disponibilizadas no contexto de Analytice II.

3.3.1 Modelagem geométrica

A modelagem geométrica é a abstração de um modelo existente no mundo real, sua forma e características físicas, como densidade e resistência, além de sua geometria. Essa modelagem é bastante utilizada em análises de dinâmica de robótica e cálculos estruturais em engenharia civil ou mecânica.

Neste trabalho, um modelo preexistente de aeronave (Figura 2) com extensão .3DS é importado. Inicialmente, nenhuma modelagem geométrica é realizada. O Analytice II não possui um módulo CAD devido à economia de recursos e também à qualidade das atuais ferramentas CAD disponíveis [10].

A ferramenta de modelagem utilizada neste projeto foi o software 3D Studio Max e o formato de arquivo adotado foi o *.3DS. A escolha desse tipo de arquivo para a criação do modelo geométrico foi determinada com base em sua estrutura, classificada como B-rep, que é representada por *meshes* e malhas de triângulos que segmentam cada face do sólido. Linhas são reconhecidas como sólidos de única face contendo dois vértices, utilizados na modelagem cinemática como eixos de rotação e translação, que será explicado a seguir.

A escolha do 3D Studio como definição do modelo foi determinada pela autossuficiência da simulação e animação existente na modelagem desse tipo de arquivo, principalmente pelo controle das condições especiais, por exemplo, a taxa de *frame-rate* [10]. Futuramente, novos formatos poderão ser inseridos.

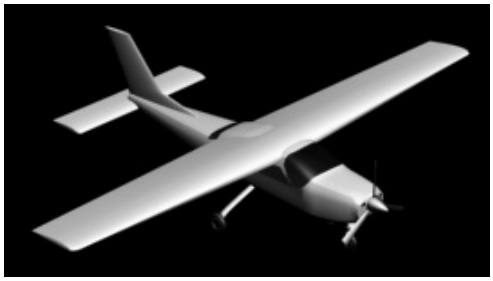

Figura 2. Modelo geométrico da aeronave

3.3.2 Modelagem cinemática

A cinemática trata da movimentação de objetos e seu posicionamento desconsiderando a dinâmica [15]. O modelo cinemático, também denominado na literatura como "modelo hierárquico" ou "de animação", relaciona as partições do modelo geométrico a processos do modelo comportamental atualizando os valores das variáveis geométricas codificadas no modelo comportamental. Na robótica, a cinemática é enfatizada como "o estudo da geometria dos movimentos de manipuladores" [16].

O uso de robôs não faz parte deste trabalho, porém alguns dos conceitos aplicados na robótica estarão presentes durante o processo de modelagem cinemática. Neste sentido, para definir os graus de movimento de um modelo geométrico duas transformações podem ser implementadas mecanicamente [17]: a rotação, definida por meio de eixos de revolução, e a translação, executada com o uso de eixos prismáticos. A Figura 3 mostra os movimentos de rotação de um modelo de aeronave.

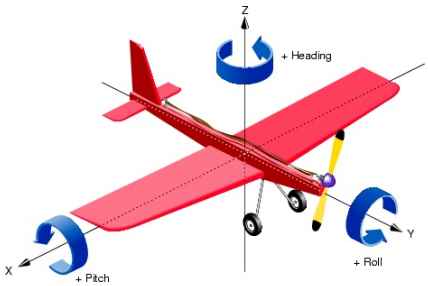

Figura 3. Ângulos de orientação RPY (roll, pitch, yaw). FONTE: [18]

Para representar a cinemática foi utilizada uma estrutura de dados em árvore onde o modelo geométrico é dividido em segmentos. Cada segmento móvel recebe uma escala angular e direção que delimita sua mobilidade. Os nós cinemáticos da árvore representam os segmentos e suas juntas; cada junta corresponde a uma variável do modelo comportamental que é utilizada na animação.

Para a construção da árvore cinemática e definição dos limites cinemáticos, é utilizado o *software* KEnvironment [10], desenvolvido no mesmo grupo de pesquisa da UTFPR.

3.3.3 Modelagem comportamental

A última modelagem realizada é a modelagem comportamental, que descreve os processos associados ao modelo e define sua interface. É responsável pela comunicação do modelo com o simulador e pela maneira como o modelo irá se comportar, como qual o grau de rotação que a aeronave deve ter ao realizar uma rolagem, por exemplo. O que define esses movimentos é um arquivo de extensão em um formato próprio *.KKT (*kinematics tree*). A ferramenta que gera esses arquivos tem a função de importar arquivos de geometria *.3DS e definir a cinemática dos modelos, entre outros dados necessários para a animação gráfica (ver Figura 4).

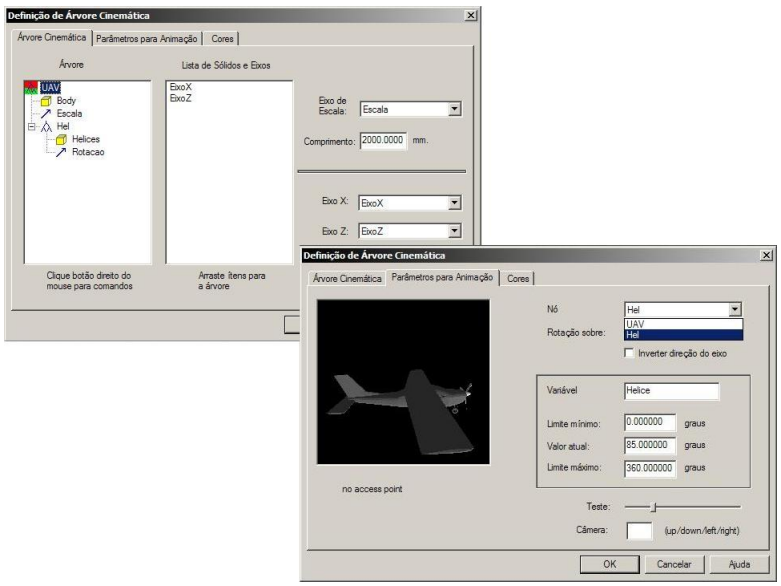

Figura 4. KEnvironment (ferramenta auxiliar para modelagem cinemática)

Dentro de arquivos de *script* haverá toda a codificação em C++ do comportamento da aeronave no cenário 3D do simulador. As classes desenvolvidas nessa modelagem são compiladas em uma DLL (Dynamic-link library) e importadas pelo Analytice II para que seja realizada a animação. A comunicação entre o modelo e o simulador é realizada por meio de macros, utilizando informações de variáveis, comandos, respostas, parâmetros e processos.

O modelo comportamental possui a seguinte estrutura:

- Processos: ativados por comandos, representam qualquer operação realizada pelo modelo, como rotação da hélice, por exemplo, consumindo tempo de simulação.
- Variáveis internas: são auxiliares e não pertinentes ao observador.
- Sinais de interação: representam a comunicação entre os modelos.

3.3.4 Desenvolvimento do modelo comportamental

Após as macros terem sido definidas, é desenvolvida uma classe na linguagem de programação C++ para o modelo, que inclui comandos de comunicação entre o modelo e o simulador, além de seus próprios atributos e comportamento. Essa classe define elementos que são comuns e essenciais em todos os modelos, tais como interface de comunicação com o *kernel*, estruturas de dados que descrevem as características de cada modelo e funções que invocam a execução de seus procedimentos.

O simulador faz uso das estruturas de dados que definem as características dos modelos e apresenta uma interface para que o usuário possa interagir com o modelo. Define também a posição dos pontos de interação física e a mudança que ocorre nesses, bem como sua relação com a execução dos processos.

Não há funcionamento do modelo sem a modelagem de sua geometria ou cinemática. Para isso, com o uso de softwares CAD cria-se o modelo geométrico já com a definição de seus eixos (rotação e/ou translação). O arquivo gerado serve como entrada para um software de auxílio que filtra e converte os dados para uma representação interna que define sua modelagem cinemática, para que assim possa ser aceito pelo animador do modelo.

Os modelos são compilados em DLLs, nas quais, por referência, estão todos os comandos e processos relativos à aeronave tridimensional e sua interação com o espaço 3D do simulador.

O controle (usuário operando o simulador em modo interativo) por meio de uma rede de Petri [19] envia ao modelo sinais de comando que são responsáveis por executar os processos operacionais do modelo, os qual pode,

por fim, enviar sinais de resposta ao simulador com a finalidade de concluir cada um desses processos. O modelo contém a completa descrição de seus processos e parâmetros operacionais.

Os processos constituem as atividades realizadas pelos modelos que demandam recursos e tempo e estão relacionadas com os sinais de comando enviados pelo controle. Os parâmetros operacionais representam características estáticas e dinâmicas dos modelos, como dimensões e capacidades, e velocidade de simulação e tempo, respectivamente.

Quando um comando dispara um processo, na sua execução ocorre a atualização das variáveis geométricas do modelo que indicam a posição dos elementos móveis na simulação para que, então, o módulo de animação gráfica, junto com o ambiente físico do simulador, determine a posição dos sólidos geométricos durante a geração de novos quadros de simulação por meio da consulta dessas variáveis.

A modelagem geométrica do modelo representa sua forma física, a posição de seus sólidos e sua escala e orientação espacial, ao passo que a modelagem cinemática define operações de rotação e translação dos sólidos e sua associação com as variáveis geométricas. Trechos de códigos pertinentes à modelagem comportamental do modelo podem ser vistos nos códigos fonte a seguir.

Biblioteca da aeronave (Aircraft.h):

```
1  #define COMANDOS 5 //Número de comandos do modelo
2  #define RESPOSTAS 4 //Número de respostas do modelo
3  #define PROCESSOS 4 //Número de processos do modelo
3  ...
4  // Sinais de Comando
5  #define LIGAR_MOTOR 0x00 //Liga o motor da aeronave
6  #define DESLIGAR_MOTOR 0x01 //Desliga o motor da aeronave
8  #define MOVIMENTAR 0x02 //Movimenta a aeronave
9  #define DECOLAR 0x03 //Decola a aeronave
10 ...
11 // Sinais de Resposta
12 #define FIM_LIGAR_MOTOR 0x00 //Ao comando LIGAR_MOTOR
13 #define FIM_MOVIMENTAR 0x01 //Ao comando MOVIMENTAR
14 #define FIM_DECOLAR 0x02 //Ao comando DECOLAR
15 #define FIM_INTERPRETADOR 0x03 //Ao interpretador do modelo
16 ...
17 // Processos
18 #define HELICES 0x00 //Liga e desliga a hélice da aeronave
19 #define AVANCO 0x01 //Aceleração da aeronave
20 #define DECOLAGEM 0x02 //Decolagem da aeronave
21 #define MANUTENCAO 0x03 //Manutenção padrão de todos os modelos
```

Classe da aeronave (Aircraft.cpp):

```
1  //Obtém a cinemática do modelo geométrico gerada com o KE
2  const char *CAircraft::ArquivoGeometria(void){
3      return "Aircraft.kkt";}
4  ...
5  //Implementa o processo de girar a hélice da aeronave
6  void CAircraft::GirarHelices(unsigned nFlag){
7      double tempo; //Variável que controla o tempo do processo
8      //Início, atualização e término do processo
9      switch (nFlag){
10         case INICIO:
11             //Inicia o tempo do processo
12             AdicionaEvento(LIGAR_MOTOR, _hora + tempo);
13             processo[HELICES].ativo = TRUE;
14             break;
```

```
15          case ATUALIZACAO_GEOMETRICA:
16              //Atualiza a velocidade de giro da hélice
17              if (vel_helices < 500)
18                  vel_helices += 2;
19              posicao += vel_helices * _delta_hora;
20              processo[HELICES].tempo_ativo += _delta_hora;
21              break;
22          case FIM:
23              //Finaliza o processo
24              AdicionaResposta(FIM_LIGAR_MOTOR);
25              processo[HELICES].ativo = FALSE;
26              break;}}
```

3.3.5 Execução da simulação

A execução da simulação de voo da aeronave 3D ocorre em uma janela do Analytice II chamada de "animador", que é o espaço 3D do simulador, como pode ser observado na Figura 5. O animador faz uso da API gráfica OpenGL a fim de representar visualmente o funcionamento da simulação. Em comparação com o DirectX, além da facilidade de utilização, a OpenGL renderiza objetos tridimensionais com eficiência, é multiplataforma e estável. A OpenGL não trabalha com sólidos "reais", mas a partir da definição das faces desses sólidos, permitindo modelar objetos e movê-los hierarquicamente. Uma comparação entre os códigos da OpenGL e da DirectX é apresentada em [10].

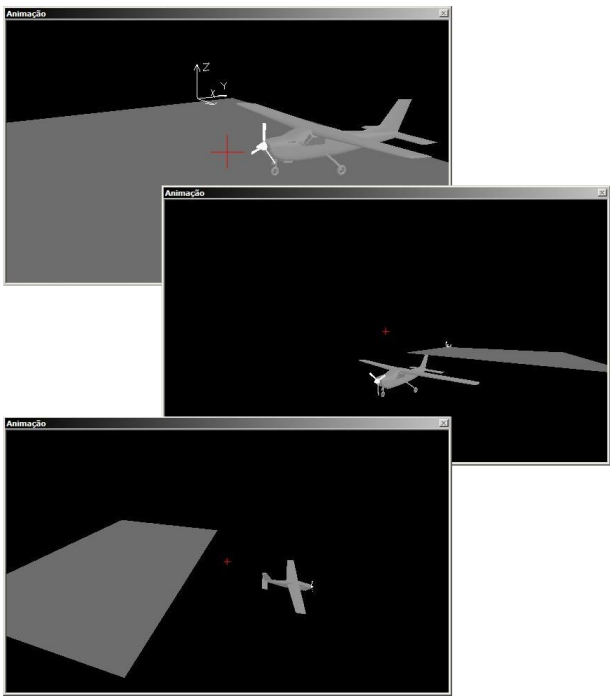

Figura 5. Modelo da aeronave no ambiente 3D do simulador

A interface de interação da simulação com o usuário é realizada por meio de uma rede de Petri, formada por comandos, respostas e pelos processos do modelo. Na Figura 6 notam-se os comandos na parte superior, os processos na parte inferior e, no centro, a atividade que corresponde à programação das atividades comportamentais do modelo.

O processo de simulação é todo manipulado e sofre interação com o usuário por meio de uma rede de Petri, como já feito anteriormente no Analytice II [10].

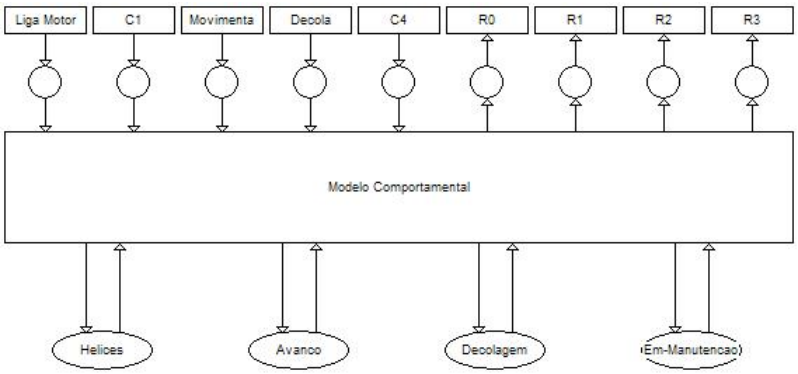

Figura 6. Interface da rede de Petri da simulação apresentando o modelo comportamental

Acerca dos resultados obtidos com a simulação, o simulador de voo foi capaz de ligar as hélices, se movimentar pelo chão do animador e decolar; por fim, foi possível realizar rolagens com a aeronave enquanto se encontrava longe do chão.

4 Conclusão

O trabalho apresentou a importância da utilização da simulação de um sistema antes de sua aplicação real, focando principalmente em simulações computacionais a eventos discretos de um simulador de voo. A ferramenta de simulação Analytice II é, em termos de engenharia e aplicação na simulação de sistemas, uma ferramenta com muitas vantagens. Sua aplicação, porém, na simulação de voo ainda necessita de algumas alterações, como, por exemplo, a substituição da rede de Petri por um controle mais interativo, como o uso de um joystick ou até mesmo o teclado na movimentação do modelo. Já a implementação manual dos eixos provê o controle sobre o modelo a ser simulado, o que possibilita uma maior compreensão do sistema pelo usuário. A utilização do driver OpenGL para a representação da animação gráfica mostrou-se mais viável do que o DirectX, além de estar aberto a uma maior portabilidade em futuras aplicações.

5 Trabalhos futuros

O desenvolvimento desse trabalho abriu novas perspectivas, como a inclusão de conceitos de aerodinâmica e utilização de frameworks gráficos mais completos, já que possuem bibliotecas de melhoramento visual e estrutura específica para aplicações tridimensionais. A estrutura criada para simular em Analytice II é original e permite o desenvolvimento de muitas aplicações. Um exemplo possível seria aplicar técnicas de reconhecimento de imagem que a aeronave iria desempenhar em uma busca territorial, assim como é realizado por alguns veículos aéreos não tripulados.

Referências

[1] BANKS J. *Discrete Event Simulation*. Prentice-Hall, Upper Saddle River, New Jersey. 2000.

[2] MATSUURA J. P. *Aplicação dos Simuladores de Voo no Desenvolvimento e Avaliação de Aeronaves e Periféricos*. CTA - ITA. 1995.

[3] *Unmanned Aircraft Systems Roadmap 2005 - 2030*. Departamento de Defesa, Estados Unidos da América. 2005.

[4] FRIGG R., HARTMANN S. *Models in Science*. Stanford. 2006.

[5] INGALLS R. G. *Introduction to Simulation*. Proceedings of the 2000 Winter Simulation Conference. 2002.

[6] KHEIR N. A. *Systems Modeling and Computer Simulation*. Marcel Dekker, Inc., New York, USA, 1988.

[7] LIN Y. B., FISHWICK P. A. *Asynchronous Parallel Discrete Event Simulation*. IEEE Transactions on Systems, Man and Cybernetics - Part A: Systems and Humans, vol. 26, n. 4, pp 397-412, july 1996.

[8] SCHRIBER T. J., BRUNNER D. T. *Inside Simulation Software: How It Works and Why It Matters*. Proceedings of the 1996 Winter Simulation Conference, ed. J. M. Charnes, D. J. Morrice, D. T. Brunner, J. J. Swain, 1996.

[9] ROSINHA L. F. F. *Proposta de Uma Arquitetura de Simulação para Sistemas Flexíveis de Manufatura e de Modelagem de Equipamentos Industriais*. UTFPR-CPGEI. 2000.

[10] KOSCIANSKI A. *Projeto e Implementação de um Simulador com Animação Gráfica para FMS*. UTFPR - CPGEI. 2000.

[11] SIMÃO J. M. *A Contribution to the Development of a HMS simulation tool and Proposition of a Meta-Model for Holonic Control.* Tese de PhD. Curso de Pós-Graduação em Engenharia Biomédica e Informática Industrial (CPGEI) no Centro Federal de Educação Tecnológica do Paraná / Universidade Tecnológica Federal do Paraná (CEFET-PR/UTFPR) e Centro de Pesquisa para Controle Automático *Nancy (CRAN)* - Universidade Henry Poincaré (UHP, França), 2005.

[12] SIMÃO J. M., KUNZLE L. A., STADZISZ P. C., SILVA P. R. O. *Atividades de Controle e Supervisão Assistidas por um Simulador de FMS*. 1º COBEF - Congresso Brasileiro de Engenharia de Fabricação, Curitiba - Brasil. 2001.

[13] SIMÃO J. M., TACLA C., STADZISZ P. C.. *Holonic Control Metamodel*. IEEE Transactions on Systems, Man and Cybernetics - Part A: Systems and Humans. Vol 39, Nº 5, Setembro. 2009.

[14] TAZZA M., KÜNZLE L. A., STADZISZ P. C., *P. C. Sistemas Flexíveis de Manufatura: Projeto e Análise*. in Anais do 9º CBA - Congresso Brasileiro de Automática, UFES - Universidade Federal do Espírito Santo, Vitória, ES, 1992.

[15] FOLEY J. D. et al. *Computer Graphics - Principles and Practice*. 2 ed., Addison-Wesley Publishing Co., USA, 1990.

[16] ASADA H., SLOTINE J. E., *Robot Analysis and Control*. John Wiley & Sons, Inc., USA, 1986.

[17] CRAIG J. J. *Introduction to Robotics, Mechanics and Control*. Addison-Wesley Publishing Company, Inc., USA, 1986.

[18] TECHPUBS Library. SGI TPL. Disponível em <http://techpubs.sgi.com/library/ dynaweb_docs/0630/SGI_Developer/books/Perf_PG/sgi_html/ch02.html>. Acesso em: 12 de maio de 2011.

[19] MURATA T. *Petri Nets: Properties, Analysis and Application*. Proceedings of IEEE, Vol. 77, Nº 4, Abril 1989.