FUNDAÇÃO DE ENSINO “EURÍPIDES SOARES DA ROCHA”
CENTRO UNIVERSITÁRIO “EURÍPIDES DE MARÍLIA” – UNIVEM
BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

BÁRBARA ANELE DE OLIVEIRA MONTEIRO

IMPLEMENTAÇÃO DE MÓDULO DE AVALIAÇÃO DA APRENDIZAGEM EM UMA FERRAMENTA VIRTUAL PARA TREINAMENTO MÉDICO

MARÍLIA
2006

BÁRBARA ANELE DE OLIVEIRA MONTEIRO

IMPLEMENTAÇÃO DE MÓDULO DE AVALIAÇÃO DA APRENDIZAGEM EM UMA FERRAMENTA VIRTUAL PARA TREINAMENTO MÉDICO

Monografia apresentada ao Centro Universitário Eurípides de Marília, mantido pela Fundação de Ensino Eurípides Soares da Rocha, como parte dos requisitos para obtenção do Título de Bacharel em Ciência da Computação.

Orientadora:
Prof[a]. Dr[a]. Fátima de Lourdes dos Santos Nunes Marques.

MARÍLIA
2006

AGRADECIMENTOS

Agradeço primeiramente a Deus por ter me dado saúde e disposição para estudar e desenvolver este projeto.

Aos meus pais Nelson e Maria Elena, e meu irmão Breno por todo amor e pelo apoio que sempre me deram não só nestes quatro anos, mas durante toda minha vida.

A todos da minha família que também me deram muito apoio e entenderam minha ausência.

A minha orientadora Fátima, por ter me incentivado sempre, me ajudado neste trabalho e mesmo sem saber, muitas vezes com poucas palavras me fez repensar e criar forças para continuar.

Aos amigos Adriano, Ana Paula, Danilo, Mariana e Rodrigo que durante estes quatro anos foram presenças constantes em todos os momentos, que com certeza sem eles não teriam a menor graça.

Aos integrantes do LApIS, em especial as amigas Ana Cláudia, Larissa e minha super companheira de laboratório Ana Paula.

Ao amigo Rodolfo pela sua paciência em ouvir todas as minhas dúvidas e pela grande ajuda e também ao meu amigo Leo, que mesmo longe sempre se preocupou e me incentivou.

MONTEIRO, Bárbara Anele de Oliveira. Implementação de Módulo de Avaliação da Aprendizagem em uma Ferramenta Virtual para Treinamento Médico. 2006. 99 F.
Graduação em Ciência da Computação – Centro Universitário Eurípides de Marília – Fundação de Ensino Eurípides Soares da Rocha, Marília, 2006.

RESUMO

A Realidade Virtual tem sido muito utilizada para aplicações na área médica com o objetivo de ajudar estudantes de medicina e profissionais de saúde no treinamento de procedimentos médicos antes de o fazerem em pacientes reais. A avaliação da aprendizagem do usuário é uma etapa muito importante em qualquer processo educacional, incluindo sistemas de treinamento médico. Neste trabalho foi desenvolvido e implementado um banco de dados e uma aplicação para auxiliarem no processo de avaliação do aprendizado do usuário em um protótipo já existente de um simulador para exame de punção de mama não imersivo, utilizando o Sistema Gerenciador de Banco de Dados Derby, a linguagem de programação Java e a API Java 3D.

Palavras-chave: Realidade Virtual, Treinamento Médico, Banco de Dados, Avaliação.

MONTEIRO, Bárbara Anele de Oliveira. Implementação de Módulo de Avaliação da Aprendizagem em uma Ferramenta Virtual para Treinamento Médico. 2006. 99 F.
Graduação em Ciência da Computação – Centro Universitário Eurípides de Marília – Fundação de Ensino Eurípides Soares da Rocha, Marília, 2006.

ABSTRACT

Virtual Reality has been largely used for applications in the medical area with the objective of helping medicine students and professionals of health in the training of medical procedures before making them in real patients. The user's evaluation is a very important stage in any educational process, including systems of medical training. In this work a database and applications were developed and implemented to aid in the process of user's evaluation in an already existent prototype of a not immersive simulator for biopsy of breast, using the Derby Database Management System, the programming language Java and the API Java 3D.

Keywords: Virtual Reality, Medical Training, Database, Evaluation.

MONTEIRO, Bárbara Anele de Oliveira. Implementação de Módulo de Avaliação da Aprendizagem em uma Ferramenta Virtual para Treinamento Médico. 2006. 99 F.
Graduação em Ciência da Computação – Centro Universitário Eurípides de Marília – Fundação de Ensino Eurípides Soares da Rocha, Marília, 2006.

RESUMEN

La Realidad Virtual ha sido muy usada para las aplicaciones en el área médica con el objetivo de ayudar a los estudiantes de medicina y profesionales de salud en el entrenamiento de procedimientos médicos antes de hacer en los pacientes reales. La evaluación del usuario es una fase muy importante en cualquier proceso educativo, incluso los sistemas de entrenamiento médico. En este trabajo se desarrolló y llevó a cabo una Base de Datos y aplicaciones para ayudar en el proceso de la evaluación del usuario en un prototipo ya existente de un simulador no inmersible para la biopsia de pecho, usando el Sistema Gerenciador de Base de Datos Derby, el idioma de programación Java y el API Java 3D.

Palabras-clave: Realidad Virtual, Entrenamiento Médico, Base de Dados, Evaluación.

LISTA DE ILUSTRAÇÕES

LISTA DE ABREVIATURAS

AV **–** Ambiente Virtual

IA – Inteligência Artificial

JDBC – Java Database Connectivity

JVM – *Java Virtual Machine* (Máquina Virtual Java)

LApIS – Laboratório de Aplicações de Informática em Saúde

RV – Realidade Virtual

SGBD – Sistema Gerenciador de Banco de Dados

SQL – *Structured Query Language* (Linguagem de Consulta Estruturada)

VRML – Virtual Reality Modeling Language

3D – Tridimensional

SUMÁRIO

INTRODUÇÃO

A Realidade Virtual (RV) é a uma forma avançada de interface do usuário que permite que ele tenha a sensação de estar dentro do ambiente tridimensional gerado por computador. O usuário pode explorar e modificar o Ambiente Virtual (AV) por meio de navegação, interação e imersão (KIRNER, 1996). Segundo Netto *et al.* (2002), a RV também pode ser caracterizada pela coexistência integrada da imersão, da interação e do envolvimento.

No mundo todo, a RV tem sido muito utilizada para aplicações na área médica em conseqüência do avanço de hardware e software. A junção da RV e da Medicina tem como um dos objetivos ajudar estudantes dessa área no treinamento de procedimentos médicos antes de o fazerem em pacientes reais. (SZÉKELY, 1999).

Na área médica, o câncer de mama é um dos tipos de câncer que mais mata mulheres no mundo. Se a doença for detectada em seu estágio inicial, o tratamento é facilitado. A punção de mama é um exame que pode ser feito para o diagnóstico do câncer de mama. (STOTZKA, 1998).

Em um trabalho anterior (LIMA, 2004), foi desenvolvido um protótipo de ferramenta para o treinamento de punção de mama. A partir dele será desenvolvido o projeto para avaliação do aprendizado do usuário neste treinamento.

Objetivo do Trabalho

O objetivo deste trabalho é o desenvolvimento e implementação de um banco de dados e uma aplicação para auxiliar no processo de avaliação do aprendizado do usuário.

O trabalho será incluído em uma ferramenta já existente desenvolvido por Lima (2004), que é um protótipo de simulador para exame de punção de mama não imersivo, desenvolvido em uma plataforma convencional. Utiliza somente dispositivos convencionais como mouse e monitor de vídeo para a interação, e tem como objetivo permitir que os usuários (alunos de Medicina ou profissionais recém-formados) treinem os procedimentos para realizar o exame de punção de mama.

Justificativa do Trabalho

Em aplicações de RV para treinamento médico, um módulo muito importante é a avaliação do aprendizado do usuário, para avaliar se o treinamento está adequado.

Para isto, a proposta deste projeto visa à implementação deste módulo para uma aplicação de treinamento médico para simular o exame de punção de mama, que está sendo desenvolvida no LApIS - Laboratório de Aplicações de Informática em Saúde.

Composição do Trabalho

Este trabalho está organizado em quatro capítulos além desta introdução.

No Capítulo 1 – Realidade Virtual e Avaliação do Usuário são apresentados conceitos básicos de RV, a aplicação de RV no âmbito da Medicina, alguns sistemas voltados à simulação e treinamento de procedimentos médicos e a avaliação do usuário neste tipo de sistema.

No Capítulo 2 – Implementação da Avaliação do Usuário, são apresentadas as tecnologias utilizadas, o protótipo de simulador de punção de mama, o modelo de dados para a avaliação e a metodologia utilizada no desenvolvimento do projeto.

O Capítulo 3 – Resultados e Discussões, são apresentados e discutidos os resultados dos testes executados a partir do sistema de avaliação implementado.

No Capítulo 4 – Conclusões e Trabalhos Futuros, são descritas a finalização do projeto e as propostas para trabalhos futuros.

E também são apresentadas as referências bibliográficas utilizadas no projeto.

CAPÍTULO 1 – REALIDADE VIRTUAL E AVALIAÇÃO DO USUÁRIO

Muitas ferramentas de RV têm sido desenvolvidas para o treinamento e simulação de procedimentos médicos com o intuito de auxiliar profissionais da saúde e estudantes de Medicina na aquisição de experiência antes de realizar procedimentos em pacientes reais.

Neste capítulo serão apresentados conceitos de RV, a RV na Medicina, a importância da avaliação do usuário em sistemas de treinamento e alguns projetos de simulação e treinamento médico.

1.1 Conceitos de Realidade Virtual

De uma maneira simplificada, a RV pode ser definida como a forma mais avançada de interface do usuário com computador até agora definida, onde o usuário pode realizar imersão, navegação e interação num ambiente tridimensional gerado por computador, utilizando canais multi-sensoriais (audição, tato, visão, temperatura, olfato, etc.) (KIRNER, 1996).

A grande vantagem da interface com RV é que o conhecimento intuitivo do usuário a respeito do mundo físico pode ser transferido para manipular o mundo virtual. O usuário pode utilizar dispositivos não convencionais como luvas de dados, conhecidas como *datagloves* (Figura 1) e capacete de visualização e controle (Figura 2) para apoiar esse tipo de interação. Estes dispositivos têm como objetivo dar ao usuário a impressão de que a aplicação está

funcionando no ambiente tridimensional real. Os dispositivos não convencionais permitem que o usuário explore o ambiente e manipule objetos com o uso das mãos (KIRNER, 1996).

Figura 1: 5DT Data Glove14 (VIRTUAL REALITIES, 2006).

Figura 2: AddVisor 150 (VIRTUAL REALITIES, 2006).

Netto *et al.* (2002) lembram que a imersão, a interação e o envolvimento são três idéias básicas que também podem caracterizar a RV pela coexistência.

Imersão é a sensação de se estar dentro do ambiente. A RV pode ser classificada em imersiva ou não imersiva. Em sistemas imersivos, é feito o uso de capacetes de visualização e cavernas. Nas cavernas são feitas projeções das visões nas paredes, teto e no piso. Além do fator visual, os dispositivos ligados aos outros sentidos também são importantes como som,

posicionamento automático da pessoa e dos movimentos da cabeça e controles reativos. Sistemas não imersivos, em geral utilizam apenas monitores de vídeo.

A interação é a capacidade do computador detectar as entradas do usuário e modificar o mundo virtual e as ações sobre ele.

O envolvimento está ligado com o grau de motivação que o mundo virtual proporciona ao usuário e pode ser passivo, como ler um livro ou assistir televisão, ou ativo, como a participação em um jogo (NETTO *et al.*, 2002).

1.2 Realidade Virtual na Medicina

Com o avanço da RV foi possível que muitas áreas do conhecimento fossem beneficiadas, como o entretenimento, saúde, negócios, treinamento e educação (BRAGA, 2001).

A saúde é uma das áreas mais beneficiadas com a RV e, em particular, a Medicina. Essa união pode proporcionar maior agilidade e confiabilidade aos exames médicos, procedimentos e atos cirúrgicos podem ser planejados com mais eficiência e torna viável o treinamento de estudantes de Medicina por meio de sistemas desenvolvidos baseados em RV (SZÉKELY, 1999).

Em todo o mundo a simulação cirúrgica baseada nas tecnologias de RV tem sido fonte de pesquisa com o objetivo de substituir métodos de treinamento e planejamento de procedimentos da Medicina já existentes. Estes sistemas permitem o estudo e observação de forma realista de casos específicos e novas metodologias e técnicas. Podem oferecer interação

baseada no toque de objetos ou visualização tridimensional de modelos reconstruídos a partir de imagens reais (MACHADO *et al.*, 2001; MACHADO, 2003).

Vários projetos estão em desenvolvimento para utilizar a RV e a tecnologia para planejar, simular e customizar procedimentos cirúrgicos invasivos como também minimamente invasivos (GREENLEAF, 2004).

Burdea *et al.* (1998) apresentam um simulador de exame de toque para diagnóstico de câncer de próstata. Na Figura 3 é apresentada a primeira visão do simulador, na qual o paciente virtual se encontra na posição adequada para o exame. O diagnóstico é feito por meio de um anel que fornece retorno tátil para o usuário. O resultado pode ser próstata normal (Figura 4(a)), próstata aumentada, tumor em estado inicial (Figura 4(b)) e tumor em estado avançado (Figura 4(c)).

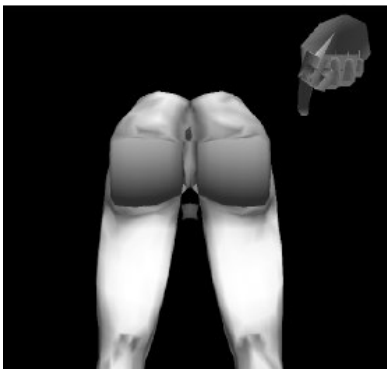
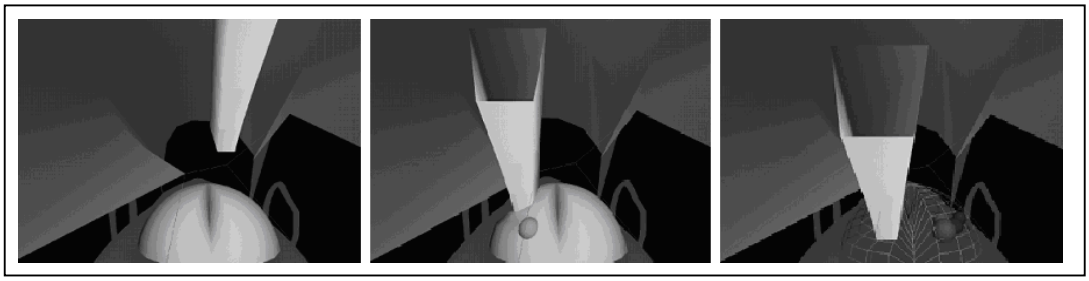


Figura 3: Primeira visão do simulador (BURDEA *et al.*, 1998).



(a) **(b)** **(c)**

Figura 4: Simulador de exame de próstata – (a) Visão interior ampliada de uma próstata normal, (b) Visão de uma próstata com tumor em estado inicial, (c) Visão de uma próstata transparente com um tumor avançado (BURDEA *et al.*, 1998).

Uma aplicação desenvolvida por Gorman *et al.* (2000) apresenta o desenvolvimento de um simulador de acupuntura lombar, que é um procedimento complexo e requer precisão da parte do médico que estiver dirigindo o procedimento. O sistema tem o objetivo de aumentar a habilidade dos estudantes e evitar erros que poderiam causar dor desnecessária e ferimentos na pele de pacientes reais durante a realização do procedimento.

No simulador, a agulha de acupuntura lombar, presa ao dispositivo háptico, penetra o dorso de um manequim humano como é mostrado na Figura 5. O dispositivo háptico fornece retorno da força, de forma que o estudante pode sentir forças resistentes ao longo de uma trajetória correta e também na incorreta. Caso incorreta, o estudante pode sentir, por exemplo, uma colisão com um osso.

Modelos 3D foram implementados em VRML para ilustrar como a agulha está sendo inserida (Figura 6).

Figura 5: Simulador de acupuntura lombar (GORMAN *et al.*, 2000).

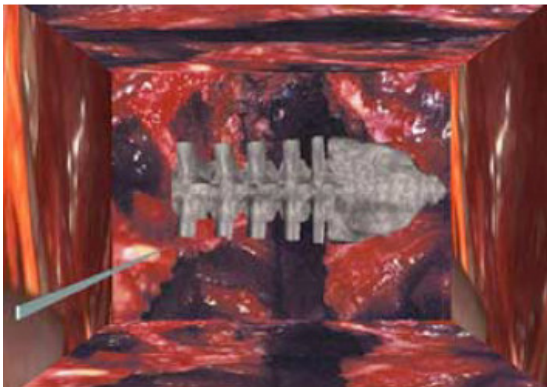


Figura 6: Modelo 3D de uma espinha dorsal em VRML (GORMAN *et al.*, 2000).

Webster *et al.* (2001) apresentam um simulador háptico de sutura (Figura 7), que incorpora componentes que modelam e deformam a pele e o material de sutura em tempo real além de fazer a gravação do estado da atividade durante a tarefa.

Utilizando este simulador, estudantes podem praticar o treinamento virtual, que pode ser em qualquer momento, e assim melhorar sua técnica (WEBSTER *et al.*, 2001).

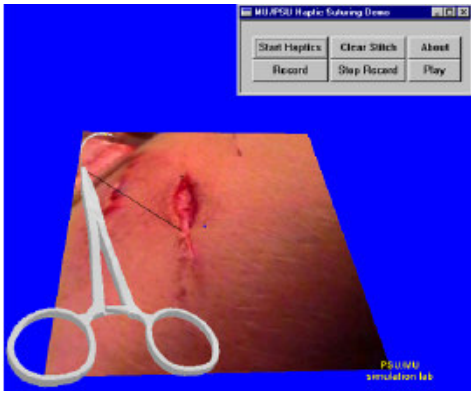


Figura 7: Sutura sendo executada no simulador (WEBSTER *et al.*, 2001).

Machado (2003) apresenta um simulador para treinamento de coleta de medula óssea (Figura 8), focado na oncologia pediátrica, sendo este o primeiro sistema de realidade virtual desenvolvido para treinamento médico no Brasil e o primeiro no mundo que trata de um procedimento pediátrico.

É um sistema de RV semi-imersivo e nele o usuário pode treinar todas as etapas envolvidas no procedimento de coleta de medula óssea. Possui quatro módulos: o módulo de localização, de coleta, de visualização interna e o de avaliação (MACHADO *et al.*, 2001).

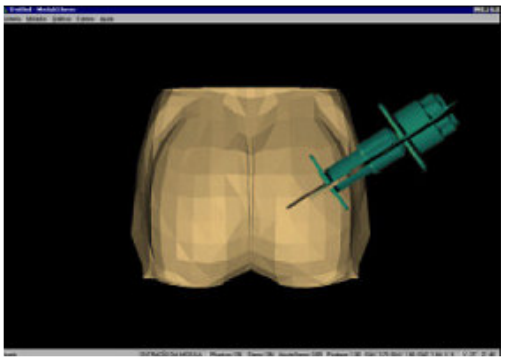
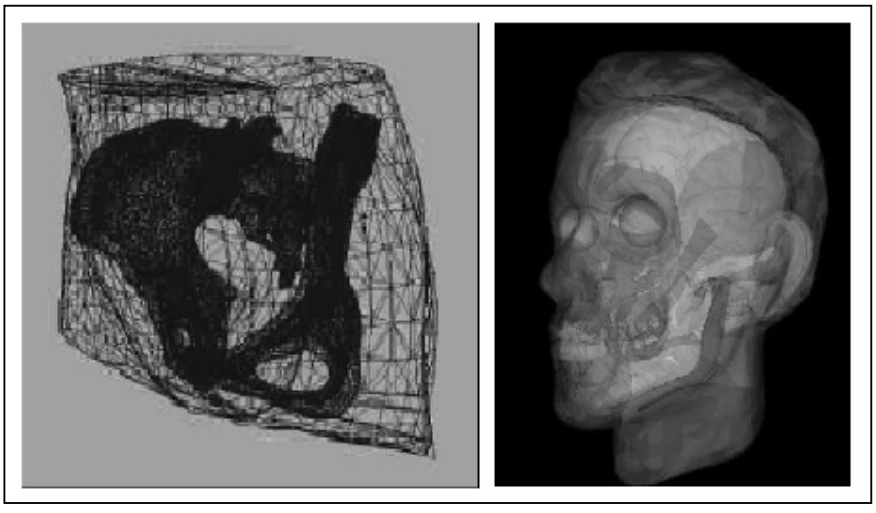


Figura 8: Tela do Simulador de Coleta de Medula Óssea (MACHADO, 2003).

Machado *et al.* (2004) apresentam o *CyberMed* que tem como objetivo apoiar o ensino e o treinamento médico por meio de explorações interativas do corpo humano e da simulação realista de procedimentos médicos em um AV imersivo.

O *CyberMed* aborda seis principais características que são: a visualização tridimensional, o uso de modelos realistas, interação especial com sensação de toque, deformação interativa das estruturas tocadas, compartilhamento visual, supervisão e avaliação das ações do usuário.

Segundo Machado *et al.* (2004) o sistema tem sido utilizado com sucesso na visualização interativa de estruturas do corpo humano, e permite ao usuário observar detalhes da estrutura real, como por exemplo, na Figura 9(a). O sistema também possibilita a observação das camadas que compõem o objeto virtual com semi-transparência como na Figura 9(b). A Figura 10 mostra os usuários utilizando o sistema para o estudo da região pélvica.



(a) **(b)**

Figura 9: Sistema *CyberMed* – (a) Modelo de bacia em malha triangular (b) Modelo de cabeça, mostrado em semi-transparência (MACHADO *et al.*, 2004).

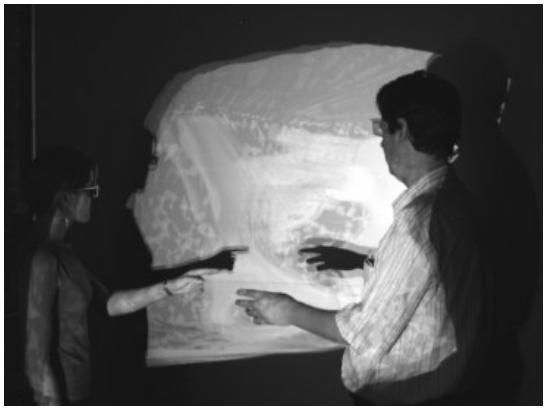


Figura 10: Sistema *CyberMed* – Estudo da região pélvica (MACHADO *et al.*, 2004).

O SITEG – Sistema Interativo de Treinamento em Exame Ginecológico é uma ferramenta de treinamento para identificar doenças relacionadas ao colo do útero desenvolvida por Souza *et al.* (2006). Segundo estimativas, este tipo de câncer é a terceira neoplasia maligna mais comum entre as mulheres.

Na ferramenta, o usuário pode simular um treinamento de forma interativa e ser avaliado na execução de um exame ginecológico do colo do útero. Os casos em que o SITEG oferece suporte são colo normal, Herpes e HPV. A interação em tempo real é feita a partir de dispositivos específicos e através de imagens estereoscópicas é conseguida a visualização.

Os autores explicam que o sistema oferece a simulação do exame de acordo com as etapas reais. Também oferece um sistema de ajuda, que disponibiliza uma série de informações sobre as patologias, explica o objetivo de cada etapa e os modos de interação (Figura 11). Um módulo de avaliação foi incorporado, com o objetivo de avaliar e monitorar as ações do usuário, além de classificar seu desempenho no treinamento.

A Figura 12 mostra a visualização na execução de um exame de toque que é a genitália feminina.

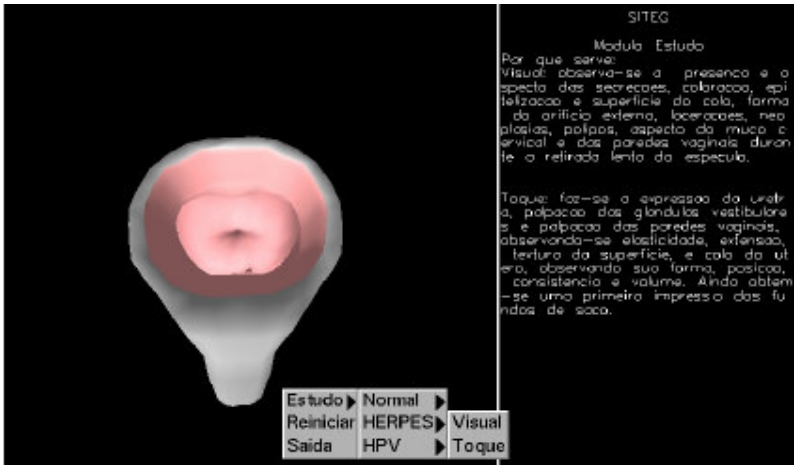


Figura 11: Menus disponíveis no SITEG, imagens de casos disponíveis e o sistema de ajuda. (SOUZA *et al.*,2006).

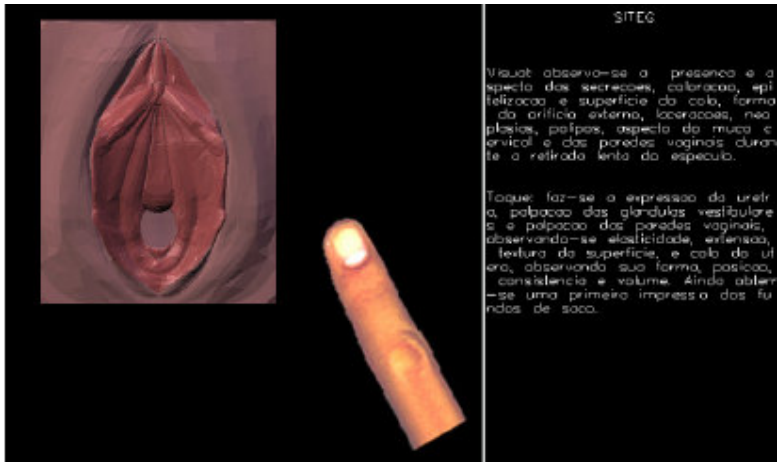


Figura 12: Execução do exame de toque (SOUZA *et al*.,2006).

Alberio *et al.* (2006) desenvolveram o ACOnTECe-Cardio, um protótipo de ambiente colaborativo para treinamento em cirurgia cardíaca que tem a finalidade de simular um transplante de coração. Além do treinamento, também é possível que o usuário apenas assista a simulação.

Existem dois tipos de cirurgia de transplante de coração: o transplante ortotópico e o heterotópico. No simulador, o transplante simulado é o ortotópico, em que o coração do paciente é retirado e em seu lugar é implantado um coração recebido de um doador.

O protótipo do ACOnTECe-Cardio é não-imersivo, executado em plataforma *desktop* e utiliza dispositivos convencionais para interação (ALBERIO *et al*., 2006).

Segundo Alberio *et al.* (2006), o Ambiente Virtual Colaborativo possui duas camadas lógicas, uma interface de comunicação com o usuário, cuja funcionalidade é apresentar graficamente o ambiente ao usuário e outra de comunicação, que é responsável pela troca de informações entre os usuários. A Figura 13 mostra a arquitetura do sistema.

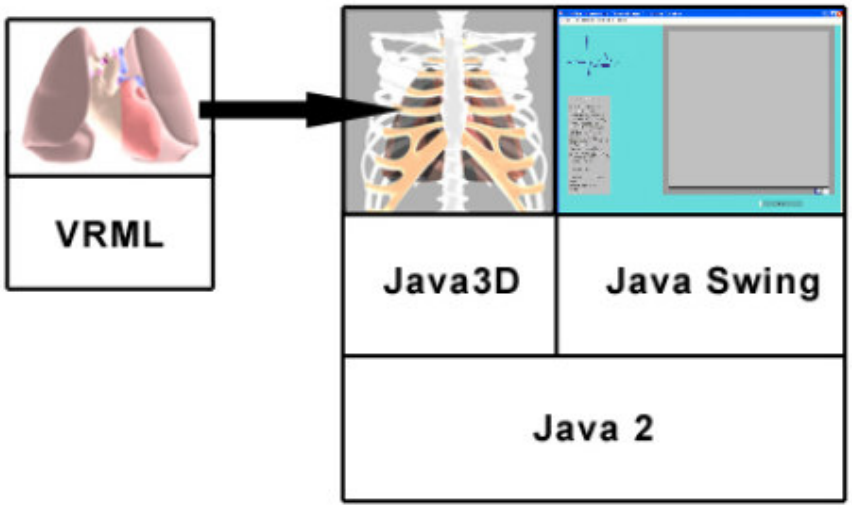


Figura 13: Arquitetura do Sistema ACOnTECe-Cardio (ALBERIO *et al*.,2006).

O sistema possui duas funcionalidades, divididas em módulo anatômico e módulo de treinamento. O módulo anatômico possui uma ferramenta de ensino da anatomia cardíaca que, além de disponibilizar uma visão mais realista do coração (Figura 14), também disponibiliza um dicionário que explica as características físicas e funcionalidades referentes a cada estrutura cardíaca (Figura 15). No módulo de treinamento é encontrado um simulador para cirurgia cardíaca. São carregados modelos tridimensionais do coração, pulmão e tórax (Figura 16). Cada participante exerce uma função no treinamento e todos eles têm a mesma visualização do ambiente, como se estivessem fisicamente em uma sala de cirurgia. Também

têm a possibilidade de se comunicarem e discutirem sobre o procedimento, através de áudio, auxiliados pelo software Skype. (ALBERIO *et al*., 2006).

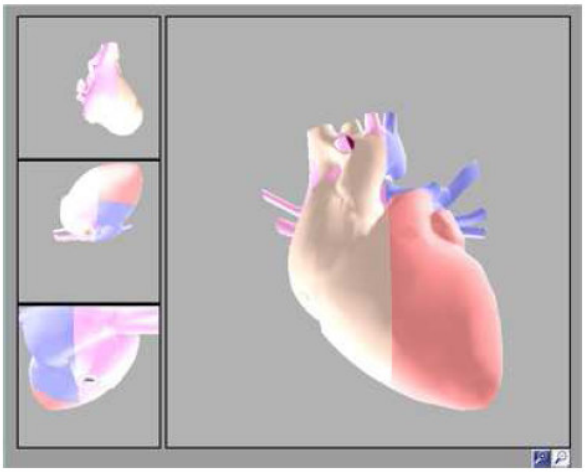


Figura 14: Visões principal e secundária da ferramenta de ensino de anatomia cardíaca (ALBERIO *et al*.,2006).

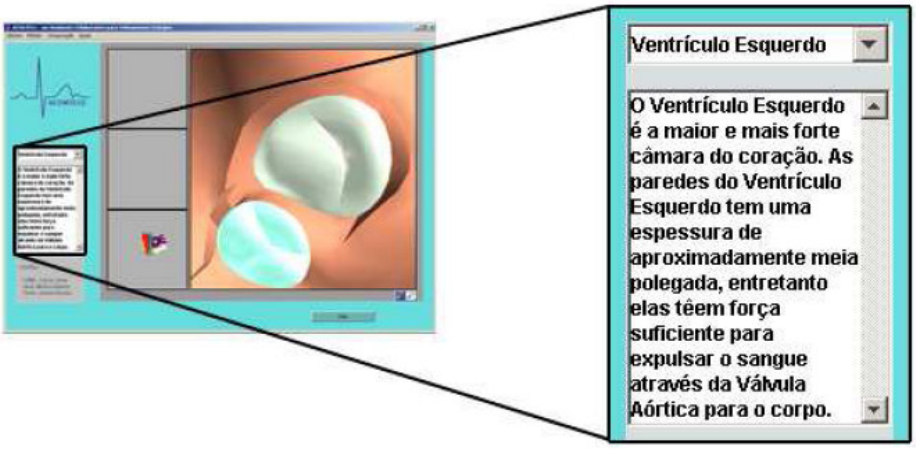


Figura 15: Dicionário explicativo da ferramenta (ALBERIO *et al*.,2006).

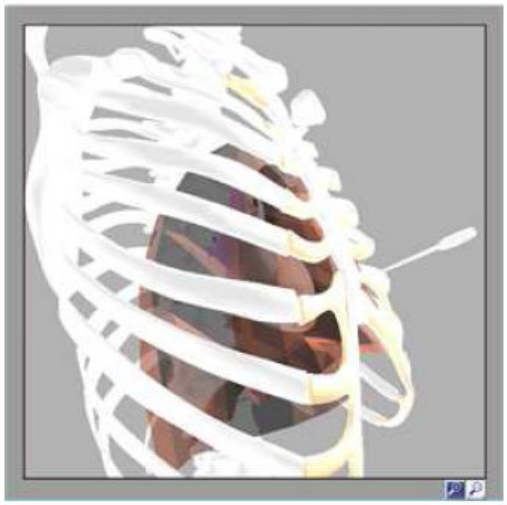


Figura 16: Treinamento interno no sistema (ALBERIO *et al*.,2006).

1.3 Avaliação do usuário e aprendizagem em sistemas de treinamento

Segundo Passerino *et al. (*2000), em qualquer processo educacional a avaliação do aprendizado é uma etapa essencial e o foco dessa etapa é a aprendizagem. A avaliação envolve aspectos de interação entre os participantes, que são os alunos e professores. É importante considerar diversas variáveis. Uma das mais importantes é o perfil do aluno. Isto é percebido pelas escolhas na forma de construir seus conhecimentos. Por exemplo, um aluno pode realizar experiências no ambiente armazenando seus dados e conclusões num banco de dados particular, ou compartilhar essas experiências com outros participantes, ou até mesmo se envolver numa construção cooperativa de uma experiência coletiva, mas mantendo sua individualidade.

Assim como em qualquer processo educacional, nos sistemas de treinamento médico a avaliação é uma etapa essencial. Na maioria dos sistemas de treinamento pesquisados na

literatura não são abordados módulos de avaliação do usuário. Os que possuem este módulo utilizam, na sua maioria, a Inteligência Artificial (IA) para o processo de avaliação.

De acordo com procedimentos já existentes ou em novos procedimentos, todo treinamento baseia-se na necessidade de uma melhora no desempenho e atuação. O processo de determinar se as mudanças no desempenho do treinamento estão de acordo com o planejado é a avaliação, que é fundamentada na medição. A medição torna necessário encontrar um meio de garantir que o que se imagina que acontece, realmente aconteça, envolvendo também o julgamento de valores (SILVA *et al.*, 2004).

Abordando a proposta apresentada neste projeto, o módulo de avaliação do sistema de Machado *et al.* (2001), citado anteriormente, classifica o desempenho do usuário que executa o procedimento. Obtidos por meio do dispositivo de reação tátil, os valores armazenados pelo simulador são força aplicada, ângulos e posição espacial, que serão utilizados em todos os módulos, inclusive no módulo de avaliação, fornecendo ao usuário a sua classificação após a prática do treinamento.

Não ligado à Medicina, mas relacionado ao escopo deste projeto o QuestComp de Abrão *et al*. (2003) , é uma ferramenta via Web para fixação e avaliação da aprendizagem através de testes e exercícios. Essa ferramenta é utilizada na PUC Minas em algumas disciplinas do curso de Ciência da Computação e fornece um recurso extra de aprendizado. O ambiente de execução da ferramenta é ilustrado na Figura 17.

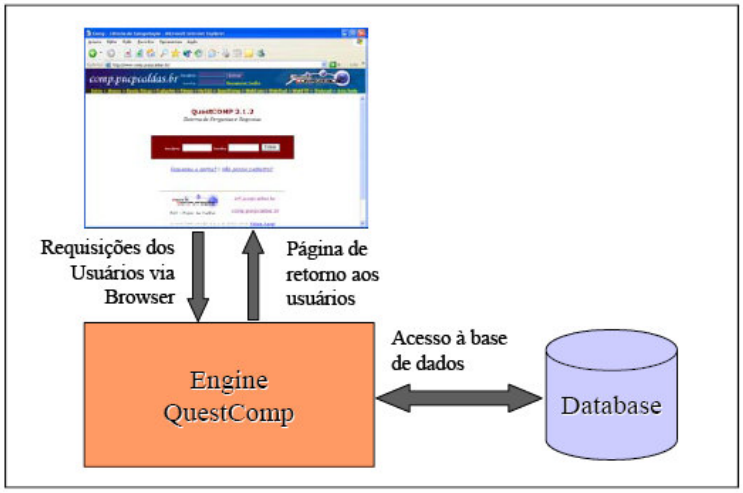


Figura 17: Ambiente de execução do QuestComp (ABRÃO *et al.*, 2004).

Segundo Abrão *et al.* (2003), a linguagem de programação utilizada na implementação do Engine QuestComp foi PHP e o Sistema Gerenciador de Banco de Dados (SGBD) MySql.

Para ter acesso ao Engine QuestComp, o usuário aluno fornece sua identificação e senha. Sua requisição é enviada ao Engine QuestComp, que para saber se é um usuário válido, utiliza consulta na base de dados e se for, retorna à página inicial, onde o usuário poderá utilizar todas as funcionalidades do sistema (ABRÃO *et al.*, 2003).

Abrão *et al.*(2003) comentam que há um módulo que se intitula Módulo Professor e também o Módulo Aluno. No Módulo Professor, os professores poderão criar provas, questionários, inserir perguntas e verificar relatórios que retornam o desempenho dos alunos. Já no Módulo Aluno, os estudantes poderão responder questionários, verificar seu desempenho nas provas e também consultar os gabaritos.

No sistema *CyberMed* de Machado *et al.* (2004), uma ferramenta de avaliação é integrada a fim de oferecer uma avaliação dos treinamentos realizados. As atividades do usuário são supervisionadas pela coleta de informações relacionadas às ações do usuário durante a interação com o sistema de treinamento. Modelos que descrevem padrões de

conhecimento de um especialista sobre os modos de realização do procedimento a ser efetuado são previamente definidos para que possam ser comparados com as informações capturadas durante a realização do treinamento.

Também não ligado à Medicina, um outro sistema de treinamento que aborda a questão de avaliação do usuário, proposto por Silva *et al.* (2004), é um sistema de treinamento de pilotos – STP, que tem como finalidade fornecer subsídios para auxiliar no desenvolvimento de sistemas de treinamento baseados em computador. Consiste em um estudo de caso para o treinamento de pilotos para um tipo específico de aeronave e abrange procedimentos de rotina e até mesmo manobras operacionais durante instabilidades.

Nesse projeto, o modelo para avaliação do treinamento que foi adotado é um modelo de MacLeod (1997), onde são abordados os seguintes aspectos:

1 – Reações do estudante: compõe o nível mais baixo da avaliação. Nesta etapa devem ser respondidas perguntas do tipo: "*Qual a importância...* e *Qual a utilidade...*";

2 – Aprendizado: esse aspecto engloba os testes de conhecimentos de procedimentos operacionais dos pilotos nos diversos sistemas;

3 – Comportamento: neste caso, do estudante após o treinamento, no desempenho normal de suas tarefas;

4 – Resultados: estes se referem aos benefícios do treinamento para a empresa, por exemplo, se após o treinamento de todos os pilotos foi detectada uma redução no tempo de espera na saída das aeronaves;

5 – Benefícios sociais: no topo dos aspectos da avaliação, os benefícios sociais relacionados ao treinamento fornecem subsídios para fortalecer a imagem da empresa.

O Sistema de Avaliação do STP conta com padrões de operações corretas pré-definidas. Então, dependendo da operação, qualquer piloto que utilizará o sistema deverá seguir certo número de passos para executar esta operação.

As interações do piloto com a cabina e também o resultado destas interações são enviados a um computador externo pela porta de comunicação serial. Depois são convertidas, interpretadas e armazenadas em um banco de dados. O tempo gasto na interação com o simulador também é armazenado. O banco de dados contém: código do piloto, código da operação, código da chave acionada e o tempo da operação em segundos. O modelo de comunicação do Sistema de Avaliação do STP pode ser melhor compreendido com a verificação da Figura 18.

Os dados armazenados durante o treinamento são utilizados para fornecer informações sobre o desempenho do piloto e ficam disponíveis após cada simulação tanto para o aluno quanto para o professor. Estas informações são visualizadas de duas formas: listagem e gráficos.

Segundo os autores, apesar de ter se mostrado satisfatória, a estratégia de avaliação utilizada no STP poderia ser disponibilizada para o piloto em tempo real para que o piloto redirecione sua lógica de atuação.

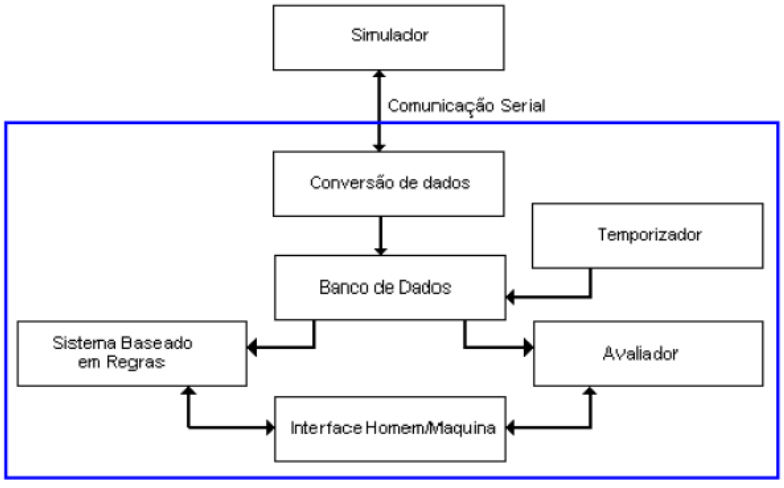


Figura 18: Modelo de comunicação do STP (SILVA *et al.*, 2004).

No sistema SITEG desenvolvido por Souza *et al.* (2006), o módulo de avaliação é implementado por uma classe, que avalia o usuário de forma objetiva a partir de informações coletadas durante a simulação. Posteriormente será desenvolvida uma classe mais complexa para que o usuário possa ser avaliado de uma forma mais subjetiva e detalhista.

Nesse módulo, o usuário pode inserir o resultado das suas impressões como forma da avaliação dele sobre o caso experimentado. Sua avaliação é comparada à patologia apresentada e o módulo de avaliação emite uma avaliação objetiva sobre o desempenho do usuário.

1.4 Considerações Finais

Neste capítulo foram apresentados conceitos de RV e algumas ferramentas de treinamento médico virtual desenvolvidas para auxiliar profissionais da saúde e estudantes de medicina a obterem conhecimento e experiência antes de executarem procedimentos médicos em pacientes reais. E, apesar da sua importância, não foi encontrado na literatura um modelo para avaliação subjetiva da aprendizagem nesse tipo de sistema que não utilizasse técnicas de Inteligência Artificial.

CAPÍTULO 2 – IMPLEMENTAÇÃO DA AVALIAÇÃO DO USUÁRIO

Neste capítulo será apresentado o protótipo de simulação de punção de mama no qual foi inserido o módulo de avaliação do usuário, o modelo de dados utilizado e também a metodologia utilizada para a avaliação do usuário no protótipo.

2.1 Tecnologias Utilizadas

Para implementação da avaliação do aprendizado usuário no protótipo de ferramenta para simulação do exame de punção de mama de Lima (2004), foi utilizada a Linguagem Java juntamente com a API Java 3D e o SGBD Derby, que é gratuito. Para a escrita do código foi utilizado o programa Eclipse na versão 3.2 e instalados alguns *plugins* que permitem acesso ao SGBD diretamente do Eclipse (ECLIPSE, 2006).

Em conseqüência do funcionamento do protótipo depender de versões específicas da linguagem Java, como J2SDK 1.4 e Java 3D 1.3, foram utilizadas as mesmas na implementação da avaliação do usuário. Porém o SGBD MySQL foi substituído pelo Derby.

O Derby é um SGBD relacional leve, utilizado para pequenas aplicações. É baseado em Java e utiliza Linguagem de Consulta Estruturada (SQL - *Structured Query Language*) padrão e *Java Database Connectivity* (JDBC). Ele não requer administração e executa na mesma Máquina Virtual Java (JVM) que o aplicativo, podendo ficar praticamente invisível ao usuário final. (APACHE, 2006).

2.2 Protótipo de Simulação de Punção de Mama

O protótipo utilizado para treinamento médico de punção de mama, desenvolvido por Lima (2004), cadastra pacientes com as imagens mamográficas do seu exame, uma com a visualização crânio-caudal e outra com a visão médio-lateral e também faz a manutenção dos pacientes já cadastrados. Após o cadastramento de pacientes, o protótipo permite ao estudante de Medicina selecionar um caso e executar o treinamento da punção. O treinamento é feito com interação do mouse para controlar a seringa.

Para iniciar o protótipo, o programa principal deve ser iniciado, escolhendo-se o idioma desejado. Depois da confirmação (Figura 19), o menu do protótipo é carregado (Figura 20).

No menu Arquivo, é encontrado o sub-menu "Manutenção de Pacientes". No menu Selecionar Caso, é possível selecionar um caso pré-definido para executar o treinamento.

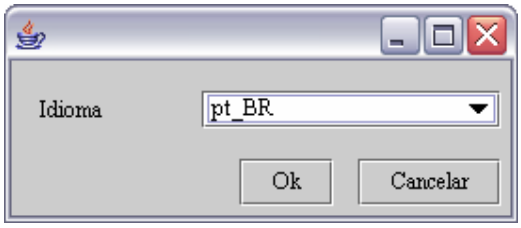


Figura 19: Tela Inicial do protótipo (LIMA, 2004).

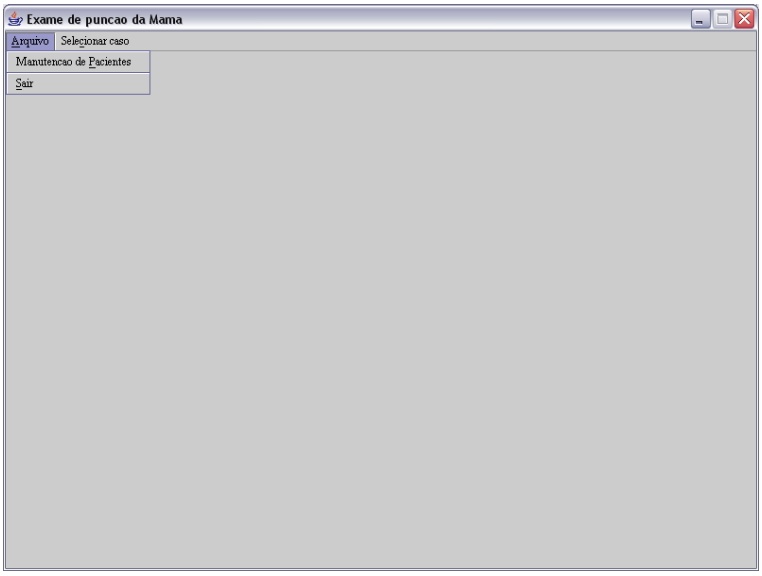


Figura 20: Menu Principal do protótipo (LIMA, 2004).

Inicialmente, no protótipo na classe `ColisaoMama.java` havia sido implementada somente a colisão na agulha da seringa com o nódulo, não mostrando a primeira colisão, que seria da agulha com a mama.

Durante o cadastro do paciente, no momento em que são carregadas as imagens mamográficas, há também campos para serem cadastrados o tamanho do nódulo observado e a posição espacial (X, Y e Z) para que o nódulo seja gerado dinamicamente, porém somente a título ilustrativo. O nódulo inserido na mama tem tamanho fixo, não possibilitando a inserção do mesmo de acordo com a observação das imagens mamográficas do paciente.

Quando selecionado o caso para executar o treinamento (Figura 21), são carregadas as imagens deste caso e também a cena 3D (Figura 22).

Na tela do exame, são encontrados dois botões: “Iniciar Punção” e “Troca Mama”. O “Iniciar Punção” permite os movimentos de vaivém do êmbolo da seringa para extrair o material de dentro do nódulo e o “Troca Mama” carrega uma outra mama na cena, que simula a deformação da mama que ocorre quando o médico pressiona a mesma.

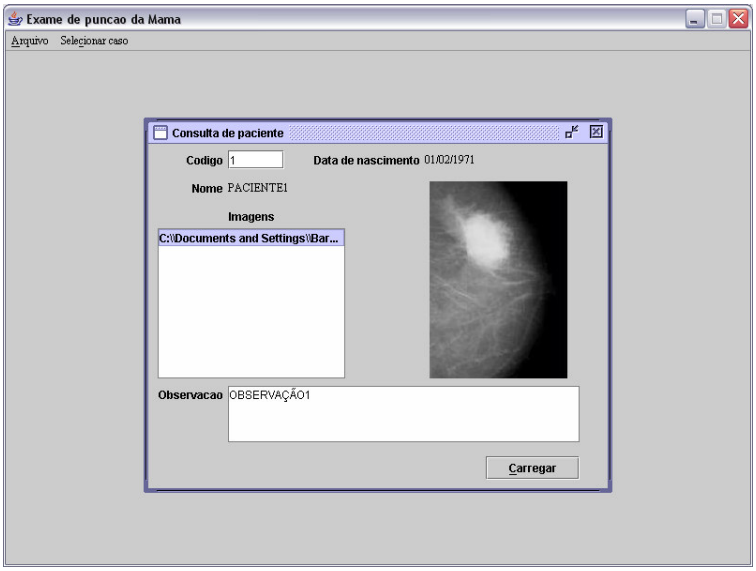


Figura 21: Seleção do caso para execução do treinamento (LIMA, 2004).

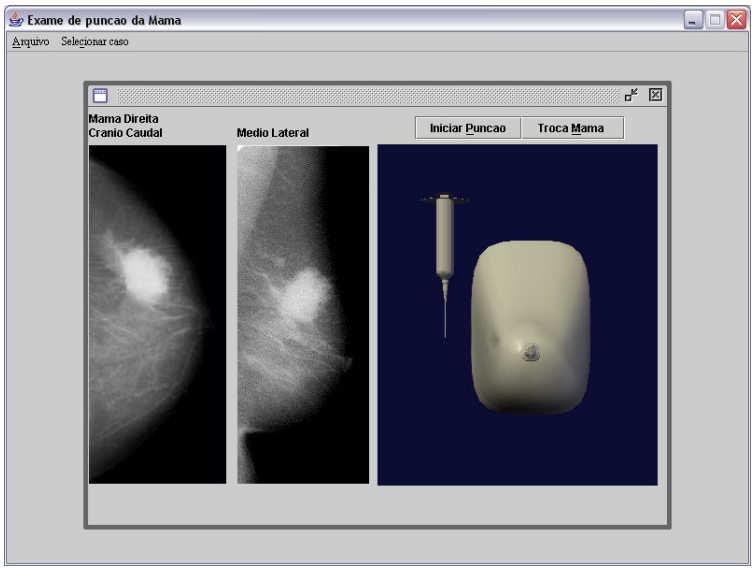


Figura 22: Tela onde será executado o treinamento (LIMA, 2004).

Com o auxílio do mouse, o estudante de medicina pode movimentar a seringa e inserir a agulha na mama na posição desejada para encontrar o nódulo e executar a punção. Assim que o nódulo é encontrado, é mostrada uma mensagem indicando que o nódulo está em contato com a agulha e o exame é finalizado, como mostrado na Figura 23.

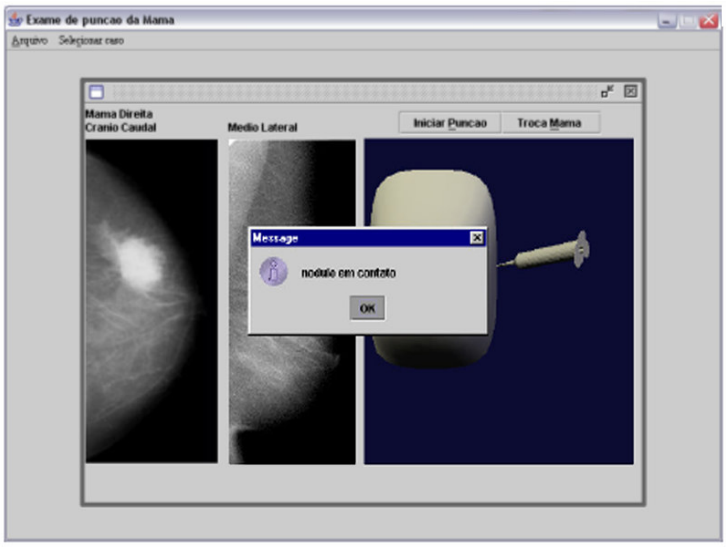


Figura 23: Exibição do treinamento finalizado (LIMA, 2004).

A Base de dados implementada no protótipo é composta por duas tabelas, sendo uma para armazenamento dos pacientes e a outra para as imagens. A Figura 24 apresenta o modelo de dados utilizado no protótipo.

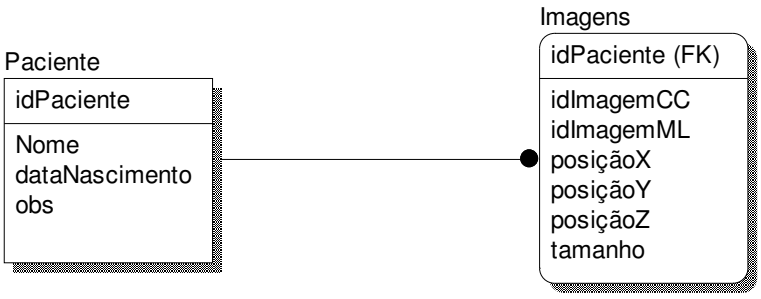


Figura 24: Modelo de Dados do protótipo para exame punção de mama.

O diagrama de classes do protótipo é apresentado na Figura 25 e nele já estão inseridas as classes `ColisaoNodulo.java` e `ColisaoMama.java` que foram implementadas neste projeto.

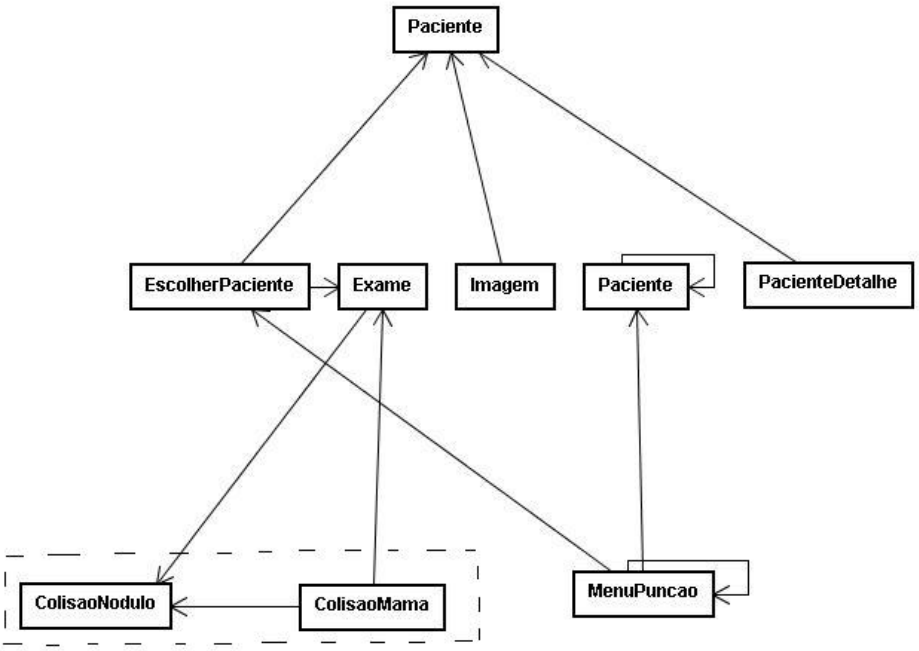


Figura 25: Diagrama de Classes

2.3 Modelo de Dados

Foi feita uma migração do Banco de Dados (BD) do protótipo de punção de mama do SGBD MySQL para o Derby, para que haja integração entre as tabelas utilizadas para manutenção e avaliação do usuário, e as tabelas pré-existentes do protótipo.

O modelo de dados criado para o módulo de avaliação, incluindo os tipos dos atributos, é apresentado na Figura 26.

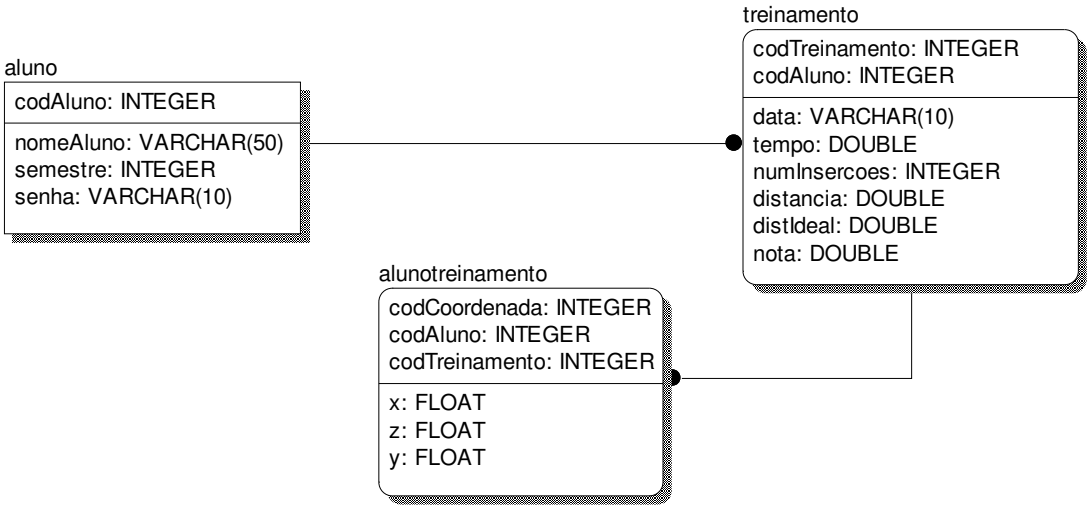


Figura 26: Modelo de dados do módulo de avaliação.

A tabela ALUNO foi criada para armazenar os dados cadastrais dos alunos que executarão o treinamento no protótipo. A tabela TREINAMENTO armazena os dados do treinamento relacionados à identificação de um aluno. Há também a tabela COORDENADA que armazena as coordenadas da colisão e da movimentação da agulha na mama e da colisão no nódulo para cada treinamento de cada aluno, a fim de fornecer parâmetros para a avaliação.

Foram criadas as classes `ConnectionManager.java` e `Aluno.java` para conexão e manipulação de dados nas tabelas da base de dados do sistema de avaliação.

A classe `ConnectionManager.java` permite a conexão da aplicação com o SGBD, apresentada no trecho de código em Java na Figura 27.

```
public static void ConnectDB (String fonte)
{
    try
    {
        String url = "jdbc:derby:" + fonte;
        Class.forName( "org.apache.derby.jdbc.EmbeddedDriver");
        con = DriverManager.getConnection(url);
        stmt = con.createStatement();
        con.setAutoCommit(true);
        System.out.println("Base de dados aberta!");
    }

    catch (Exception e)
    {
        e.printStackTrace();
        System.out.println("Problema na abertura da base de dados!");
    }
}
```

Figura 27: Trecho de código para conexão com o SGBD.

A classe `Aluno.java` contém o construtor do objeto Aluno e também os métodos para manipulação da tabela ALUNO. Na Figura 28 é apresentado o trecho de código do método que faz inserção dos dados na tabela ALUNO.

```
private boolean insertDB()
{
    try {
          String insertSQL = "INSERT INTO aluno(codAluno, nomeAluno, semestre,senha)  VALUES ("
              + codAluno + ",'" + nomeAluno + "'," + semestre + ",'" +senhaUsuario +"')";
          ConnectionManager.stmt.executeUpdate(insertSQL);
          ConnectionManager.con.commit();
          JOptionPane.showMessageDialog(null,"Usuário Cadastrado Com Sucesso");
          return true;
        }
    catch(SQLException e) {
        try {
              if (e.getErrorCode() == -193) {
              JOptionPane.showMessageDialog(null,"Já existe este usuário cadastrado!"
                      + e.getErrorCode() );
              return false;
            }
        }
        catch(Exception ex) {
            JOptionPane.showMessageDialog(null,"Problemas com Tratamento de Erros!" );
            return true;
        }
    }
  return true;
}
```

Figura 28: Trecho de código da inserção de dados na tabela aluno.

Para a manutenção da base de dados, foi criado um aplicativo que permite a inserção, atualização e remoção de alunos no sistema, que pode ser chamado no Menu Principal como Manutenção de Usuários (Figura 29). No Menu Principal (Figura 30) é executado o login, fornecendo-se o código do usuário e a senha previamente cadastrados. Somente um aluno

cadastrado poderá executar o treinamento. A verificação do código e da senha é feita por meio de uma consulta na tabela ALUNO.

Figura 29: Tela de Manutenção do usuário.

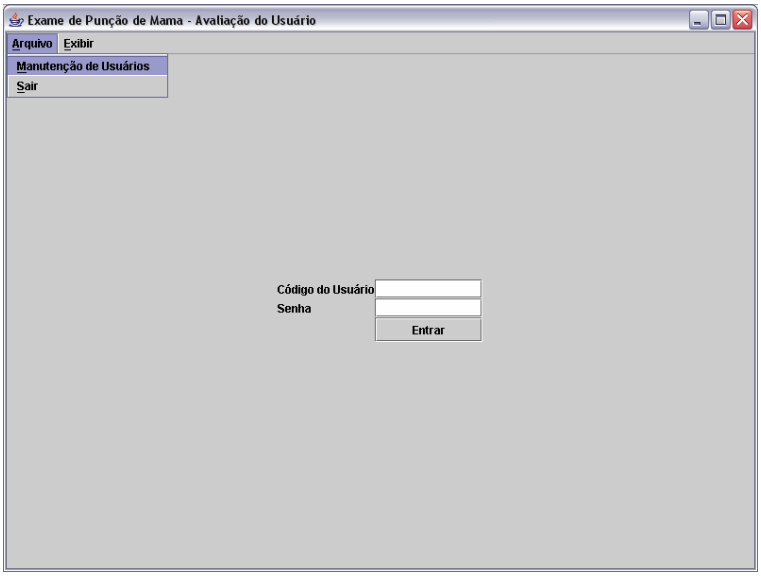


Figura 30: Tela do Menu Principal.

2.4 Avaliação da Aprendizagem

A partir do momento em que o aluno usuário do sistema está autenticado, é direcionado à tela do treinamento do exame de punção, na qual poderá fazer a manutenção de pacientes, que são os casos disponíveis para a execução do exame, e também executar o treinamento com casos pré-definidos, como na Figura 31.

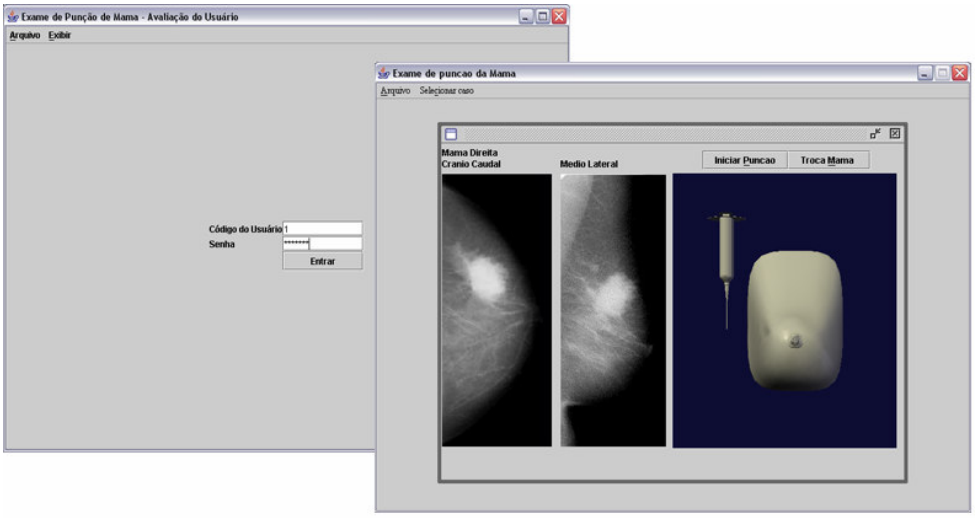


Figura 31: Tela de login e redirecionamento para o treinamento.

Para conseguir uma forma de avaliação do usuário, foram feitas algumas alterações na classe `ColisaoMama.java`, na `ColisaoNodulo.java` e também na classe `Exame.java` do protótipo existente, que estão apresentadas respectivamente no Apêndice G, Apêndice H e Apêndice I . Na classe `Exame.java`, havia sensor de colisão na agulha, e o nódulo podia ser colidido, porém a mama não podia ser colidida porque não é possível a colisão da agulha com mais de um objeto.

No exame de punção de mama, o médico insere uma agulha na mama para coletar o material para análise. Para que não cause desconforto à paciente, deve haver o mínimo de inserções, movimentações e com o menor tempo possível. Partindo destas premissas, elaborou-se o módulo de avaliação deste projeto, onde se armazena o número de vezes em que a agulha é inserida na mama, obtém-se os tempos inicial e final da execução deste exame e armazenam-se todas as posições de movimentação da agulha utilizada para medir o caminho percorrido por esta agulha, ponto a ponto, até ao final do exame. Este caminho pode ser obtido através da distância Euclidiana de um ponto a outro. Depois de coletados, estes dados podem ser analisados e comparados aos tempos, caminhos e números de inserções ideais em um exame deste tipo. Utilizando colisões entre objetos 3D, vetores, funções de tempo, fórmulas matemáticas e armazenamento e consulta à base de dados, foi possível coletar e calcular todos estes dados.

Para conseguir utilizar a distância Euclidiana (Figura 32) para traçar o caminho feito pela agulha durante a inserção da mesma, foi necessário inserir colisão nos objetos que representam a mama e o nódulo. Assim os dois objetos poderiam colidir com a agulha, apesar de nos casos reais, a agulha é quem se movimenta e colide com a mama, porém da forma implementada previamente, não era possível obter as duas colisões necessárias quando o sensor de colisão se fazia presente na agulha. A colisão na mama (Figura 33) e no nódulo (Figura 34) são inseridas no método que carrega o objeto na cena.

$$distância = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2}$$

Figura 32: Fórmula da distância Euclidiana

```
public BranchGroup setup_mama()
{
    ...

    BranchGroup b = mama.getSceneGroup();
    b.setCapability(BranchGroup.ALLOW_DETACH);

    Shape3D shape = null;
    shape = (Shape3D) b.getChild( 0 );

    ColisaoMama cd = new ColisaoMama(shape,codAluno,codTreinamento,cn,this);
    BoundingSphere boundsCollision = new BoundingSphere();
    cd.setSchedulingBounds(boundsCollision);
    b.addChild(cd);

    ...
}
```

Figura 33: Trecho de código da classe Exame.java onde é inserida colisão na mama.

```
private Group carregaNodulo()
{
    ...

    BranchGroup branchGroup = scene.getSceneGroup();
    Shape3D shape = null;
    shape = (Shape3D) branchGroup.getChild( 0 );
    ColisaoNodulo cn = new ColisaoNodulo(shape,codAluno,codTreinamento);
    BoundingSphere bounds = new BoundingSphere();
    cn.setSchedulingBounds(bounds);
    branchGroup.addChild(cn);

    ...
}
```

Figura 34: Trecho de código da classe Exame.java onde é inserida colisão no nódulo.

Na classe `ColisaoMama.java`, a partir do momento da colisão da mama com a agulha, são armazenadas as posições espaciais da colisão e de todos os movimentos da agulha em um vetor até que haja a colisão do nódulo com a agulha. Essas posições são gravadas na tabela COORDENADA, onde sempre a última posição armazenada será a posição de colisão do nódulo. Para se obter essa condição, foram feitas modificações nos métodos da classe `ColisaoMama.java` (Figura 35) e foi implementada a classe `ColisaoNodulo.java` (Figura 36) a fim de que fosse possível tratar separadamente a colisão do nódulo. Na classe `ColisaoMama.java` tratava-se a colisão da agulha com o nódulo e foi modificada para tratar a colisão do objeto mama com a agulha. Foram inseridas a forma de obter as

coordenadas das colisões, movimentações da agulha e o número de inserções da agulha na mama e também o cálculo das distâncias.

```
public class ColisaoMama extends Behavior {
        ...
                else if(o==wMovement)//se movimentou
                {
                    if(!cn.inCollision)// se o nódulo não está em colisão com a agulha
                    {
                        Vector3f v3fmama = new Vector3f();
                        SceneGraphPath scenemama = wMovement.getTriggeringPath(); //Objeto que causou a colisão
                        Transform3D transfmama = new Transform3D();
                        scenemama.getObject().getLocalToVworld(transfmama);
                        transfmama.get(v3fmama);//armazena as coordenadas de cada movimentacao no Vector3f
                        x = v3fmama.x;// obtém-se os valores da coordenada separadamente em "x"
                        y = v3fmama.y;// "y"
                        z = v3fmama.z;// e "z"
                        numMovimento ++; // incrementa o código da coordenada
                        insereCoord(); // insere a coordenada no banco
                        wakeupOn(ou); //acorda com qualquer evento
                    }
                    else{ tempo = (cn.tempofim - tempoinicio)/1000; // calcula o tempo gasto no exame
                        x = cn.x; //obtém-se os valores da coordenada separadamente em "x"
                        y = cn.y; //"y"
                        z = cn.z; //"z"
                        numMovimento ++;// incrementa o código da coordenada
                        insereCoord(); // insere a coordenada do nódulo
                        distIdeal();// calcula a distância ideal
                        distancia();// calcula a distância percorrida no exame
                        updateTreinamento();// atualiza a tabela treinamento
                        JOptionPane.showMessageDialog(null,"Nódulo em contato! Exame terminado");
                    }
                }
        ...
}// end ColisaoMama
```

Figura 35: Trecho de código da classe ColisaoMama.java.

```
public class    ColisaoNodulo   extends Behavior {
    ...

    public void processStimulus(Enumeration criteria)
    {
        while (criteria.hasMoreElements())
        {
            Object o = criteria.nextElement();

                if (o==wEnter) // se o nódulo colidiu com a agulha
                {
                    inCollision = true;
                    tempofim = System.currentTimeMillis();
                    Vector3f v3fnodulo = new Vector3f();
                    SceneGraphPath sg1 =wEnter.getTriggeringPath();//Objeto que causou a colisão
                    Transform3D t3d1 = new Transform3D();
                    sg1.getObject().getLocalToVworld(t3d1);
                    t3d1.get(v3fnodulo);//armazena a coordenada do ponto de colisão do nódulo com a agulha
                    x = v3fnodulo.x;//obtém-se os valores da coordenada separadamente em "x"
                    y = v3fnodulo.y;//"y"
                    z = v3fnodulo.z;//"z"
                    wakeupOn(wExit);//acorda se sair da colisão
                }
    ...
        }
    }
}
```

Figura 36: Trecho de código da classe ColisaoNodulo.java.

Para se obter as posições espaciais a serem gravadas, utilizou-se o método `getTriggeringPath()` da classe `WakeupOnCollisionMovement`. Este método retorna o grafo de cena da agulha, que é o objeto que colidiu com a mama. Depois de obtido o grafo de cena, é possível chegar na agulha por meio do `scenemama.getObject()`, que retorna um objeto `Node`, e depois é utilizado o método `getLocalToVwordl()` da classe `Node` para obter a posição das transformações que ocorreram na agulha, que podem ser melhor interpretados pelo trecho de código da Figura 37.

```
Vector3f v3fmama = new Vector3f();
SceneGraphPath scenemama = wMovement.getTriggeringPath();
Transform3D transfmama = new Transform3D();
scenemama.getObject().getLocalToVworld(transfmama);
transfmama.get(v3fmama);
```

Figura 37: Trecho de código da classe ColisaoMama.java onde é obtida a posição espacial dos movimentos.

Após o procedimento são calculadas as distâncias Euclidianas entre todos os pontos que descrevem o caminho feito pela agulha durante a inserção, desde o ponto de colisão da mama com a agulha, seguindo pelos pontos de movimentação da mesma, até o ponto de colisão do nódulo com a agulha. Essas distâncias são somadas obtendo-se a distância total percorrida. Para verificar quando o nódulo é colidido, o método construtor da classe `ColisaoMama.java` recebe por parâmetro um objeto do tipo `ColisaoNodulo.` Depois que a mama colide com a agulha e enquanto não houver colisão do nódulo são armazenados os valores das posições espaciais das movimentações da agulha.

Depois é calculada a distância ideal, do ponto de colisão da mama com a agulha até o ponto de colisão do nódulo com a agulha, que deveria ser de forma retilínea. Os valores das duas distâncias são comparados e o resultado é um dos parâmetros utilizados para a classificação do usuário depois de efetuado o treinamento.

Além das coordenadas citadas, são armazenados na tabela COORDENADA o código do aluno, o código do treinamento e o código da coordenada, para que um usuário possa executar mais de um treinamento e que cada treinamento possa armazenar coordenadas seqüencialmente.

Além da distância, o tempo gasto no treinamento é armazenado e comparado a um tempo médio em que o exame deveria ser feito. Para obter o tempo corrente em milisegundos, foi utilizada a função `System.currentTimeMillis()`. A data do treinamento também é armazenada na tabela TREINAMENTO, sendo obtida e formatada por meio do trecho de código da Figura 38.

```
Date data = new Date();
java.text.DateFormat sdf =new java.text.SimpleDateFormat("dd/MM/yyyy");
String dt = sdf.format(data);
```

Figura 38: Trecho de código que mostra a obtenção e formatação da data.

O número de inserções na mama é armazenado a partir de cada colisão com a mama. O número de inserções ideal é somente um devido ao fato de que este exame requer precisão. Caso o número de inserções seja maior que um o usuário já não atingirá um bom desempenho mesmo após os cálculos das distâncias e do tempo.

O treinamento é finalizado ao se obter a colisão do nódulo com a seringa. Depois de armazenado esse ponto, é feita a classificação do usuário e assim a tabela TREINAMENTO já pode ser atualizada com todos os dados do treinamento. No caso do usuário não aingir o nódulo e desistir do treinamento, não foi implementada a solução, porém pode-se conseguir o tratamento dessa situação utilizando o tempo ideal em que o exame deve ser feito. Se o tempo se esgotar e o usuário não houver atingido o nódulo, o treinamento é finalizado e será atribuída nota zero a esse usuário.

Para classificar o desempenho do usuário no treinamento, uma nota é atribuída. Esta nota é de acordo com pesos atribuídos a cada parâmetro armazenado, dependendo da escolha do usuário, que insere os pesos escolhidos na tela de exibição do relatório juntamente com o código do usuário que deseja consultar o desempenho. Por exemplo, para o número de inserções na mama, é imprescindível que o número seja apenas um, portanto, este quesito deve ter um peso maior. Assim como para o número de inserções, o usuário deverá escolher um peso para a proporção das distâncias e um peso para a proporção do tempo, que juntos devem totalizar dez pontos. Estes pesos serão escolhidos no momento em que se desejar exibir o relatório, após os treinamentos. No presente projeto, esta visão onde é possível escolher os pesos para as categorias que compõem a nota, está tanto para aluno quanto para professor, porém futuramente esta visão estará somente para o professor. Além dos pesos, é possível escolher também um tempo ideal em que se deve executar o exame.

Cada aluno poderá executar quantos treinamentos forem necessários e todos os seus dados estarão armazenados na base de dados, para que o médico professor e até mesmo o aluno possam acompanhar os resultados e verificar se houve melhora no desempenho.

Os resultados da avaliação poderão ser visualizados por meio da opção Exibir Relatório (Figura 39). Para exibição, deve ser digitado o código do aluno que se deseja verificar o desempenho. Para a exibição do relatório, foi criada uma interface com o nome de `TelaRelatório.java`, que extende a classe `JFrame`. Nela foi criado um campo para receber o código do aluno a ser consultado, um botão e um `TextArea`, onde será exibido o resultado, por meio de consulta na base de dados.

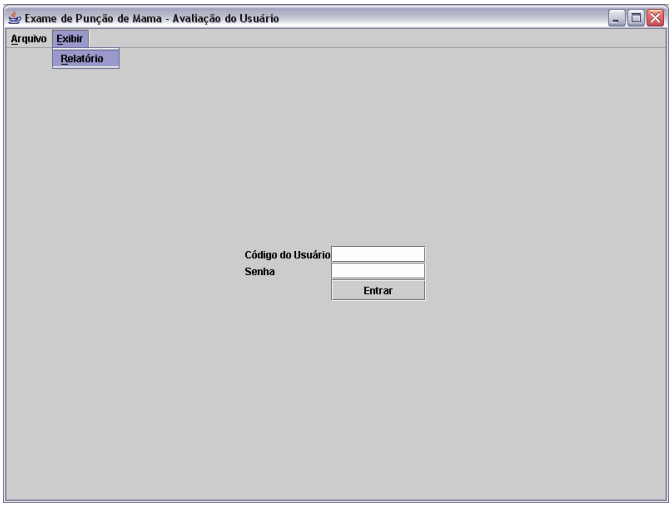


Figura 39: Tela do Menu Principal onde há o item Exibir Relatório.

O relatório mostrado na Figura 40 contém todos os dados cadastrais do aluno requerido, como código, nome e semestre em que está matriculado, todos os dados de cada treinamento efetuado por ele, como o código do treinamento, a data de cada um, o tempo gasto, os valores do caminho ideal e do caminho obtido, que são calculados pela distância Euclidiana, o números de inserções na mama, a classificação obtida e a média geral de todos os treinamentos.

Esta classificação é de acordo com uma nota Figura 41 que é calculada da forma explicada a seguir: primeiramente divide-se o tempo ideal de execução do exame pelo tempo gasto e multiplica-se o resultado pelo peso escolhido. Quando o tempo gasto é menor que o ideal, substitui-se o tempo gasto pelo tempo ideal na fórmula no momento do cálculo. Em seguida, divide-se a distância ideal, do ponto de colisão da mama até o ponto de colisão com o nódulo, pela distância obtida no treinamento e multiplica-se este resultado pelo peso escolhido anteriormente. Para o cálculo da nota quanto ao número de inserções da agulha, divide-se o número de inserções ideal, obrigatoriamente uma, pelo número de inserções no exame e multiplicando este resultado pelo peso atribuído. Após estes cálculos terem sido executados, a nota final é conseguida pela soma das três notas. Por exemplo, o usuário gastou dezenove

segundos na execução do treinamento, a distância percorrida é de 0.48977413606296366 e inseriu-se a agulha duas vezes na mama, sendo que a distância ideal seria de 0.35678419503208614. Sendo o tempo ideal vinte segundos, o peso para o tempo três, o peso para a distância três também e o peso para o número de inserções quarto, sua nota final é sete (7,0). No final do relatório, é apresentada uma média das notas de todos os treinamentos executados por este aluno.

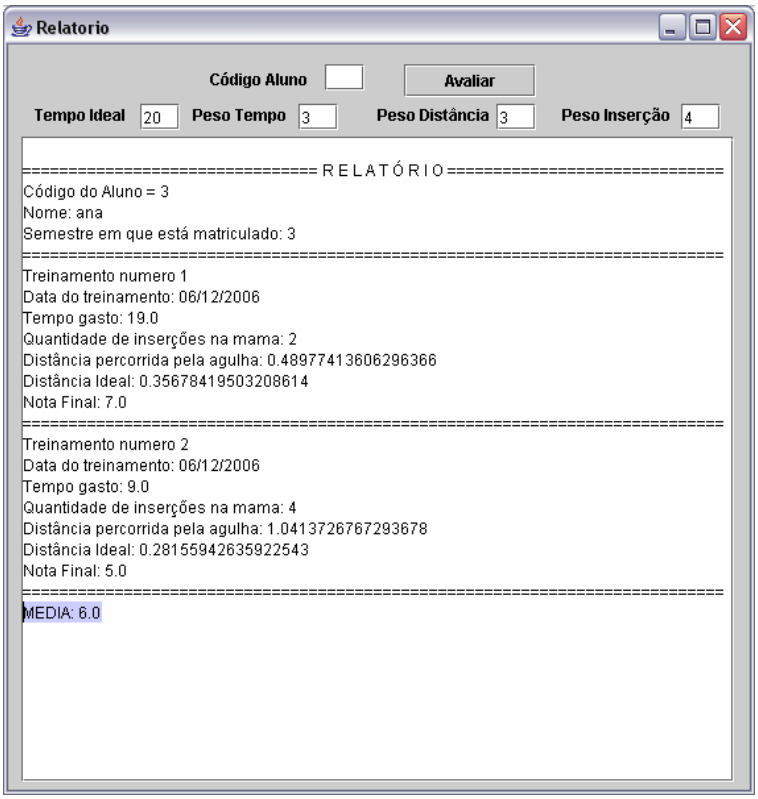


Figura 40: Tela de exibição do relatório

$$nota = \left(\frac{tempoIdeal}{tempo}\right) \cdot pesoTempo + \left(\frac{distânciaIdeal}{distância}\right) \cdot pesoDistância + \left(\frac{1}{numeroInserções}\right) \cdot pesoInserção$$

Figura 41: Cálculo da Nota.

2.5 Considerações Finais

Este capítulo descreveu o módulo implementado que avalia o aprendizado do usuário por meio dos dados obtidos durante a execução do treinamento. Apesar de poucos parâmetros, já foi possível integrá-lo a um sistema de treinamento virtual de punção de mama utilizando-o como módulo de avaliação do aprendizado dos usuários desse sistema. Os resultados obtidos a partir da implementação realizadas são apresentados no próximo capítulo.

CAPÍTULO 3 – RESULTADOS E DISCUSSÕES

Neste capítulo estão descritos os resultados da implementação da avaliação do usuário, obtidos por meio da avaliação da ferramenta integrada ao protótipo de simulação de exame de punção de mama (LIMA, 2004).

3.1 Avaliação de Usuários

Para avaliar o funcionamento do sistema de treinamento, foi feita uma pesquisa com dez pessoas, estudantes de Computação que possuem conhecimento em RV e alguns também desenvolvem aplicações de RV. Estas pessoas executaram o sistema de treinamento de punção e visualizaram o resultado obtido no relatório. Para armazenar os dados da pesquisa foi confeccionado um questionário que foi respondido ao término do treinamento.

O questionário contém perguntas para identificação do usuário e também perguntas para avaliar o módulo de avaliação de aprendizagem. Além das perguntas objetivas há um espaço onde o usuário pode fazer comentários e sugestões para melhoria do sistema. Este questionário encontra-se no Apêndice A.

A Figura 42 representa a porcentagem de facilidade de entendimento do sistema de avaliação pelos usuários que fizeram o treinamento. Verifica-se que 70% dos usuários compreenderam bem o sistema, 20% compreenderam-no plenamente e apenas 10% não o entenderam claramente.

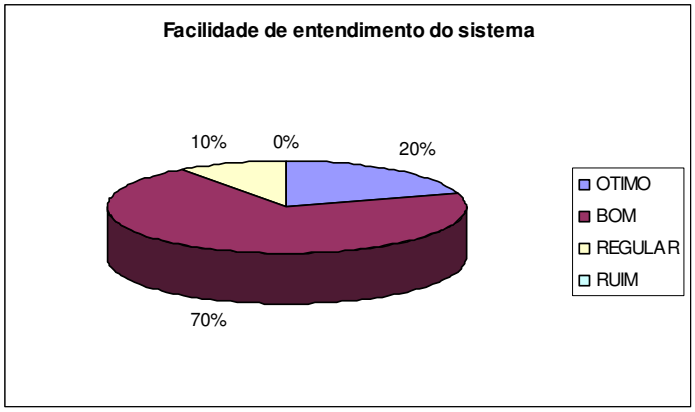


Figura 42: Gráfico que representa a facilidade de entendimento do sistema.

Na Figura 43 é representada a opinião dos usuários em relação à facilidade em acessar as funções do sistema. 30% destes usuários classificaram como extremamente fácil o acesso às funcionalidades do sistema, unindo-se a outros 60% que o classificaram como fácil. Os 10% restantes não classificaram como fácil o acesso às mesmas porque não haviam compreendido bem o sistema.

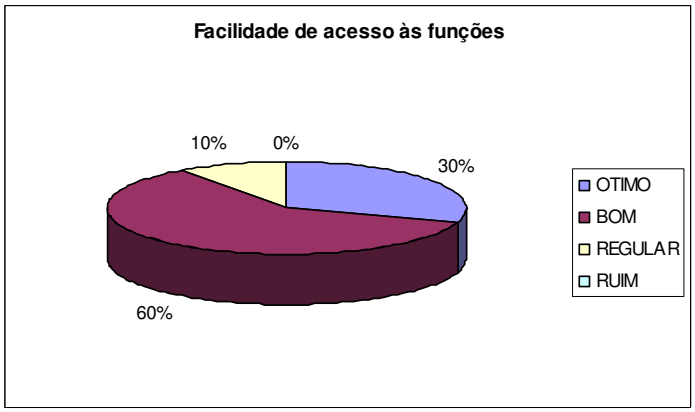


Figura 43:Gráfico que representa a facilidade de acesso às funções.

A facilidade de interpretação dos resultados exibidos no relatório foi de 80%, unidos a mais 10% que interpretaram bem os resultados e seguidos de somente 10% dos usuários que

classificaram como regular a facilidade de interpretação, de acordo com o gráfico da Figura 44.

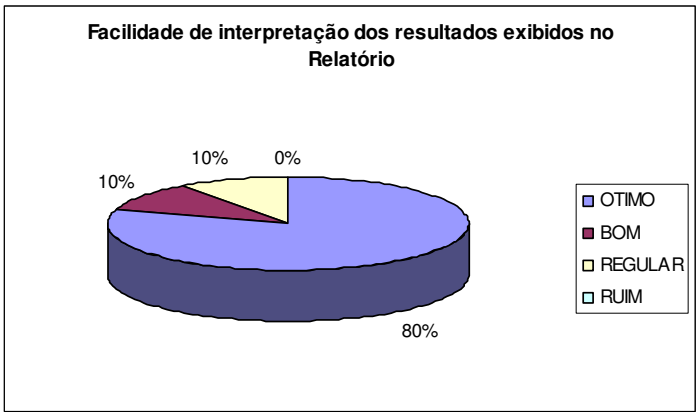


Figura 44: Gráfico que representa a facilidade de interpretação dos resultados exibidos no relatório.

Na Figura 45 está representada a porcentagem de aceitação da ferramenta de avaliação, verificando-se que 80% consideram o sistema apto para avaliar o usuário que executa o treinamento e os 20% restantes não o consideram apto porque no módulo implementado o usuário ainda pode escolher os pesos para obtenção da sua nota.

Para complementar os resultados e verificar o que pode ser melhorado, detectando falhas, além dos testes feitos com usuários envolvidos no desenvolvimento de ferramentas de treinamento médico, a pesquisa também deverá ser feita com estudantes de medicina e profissionais da área.

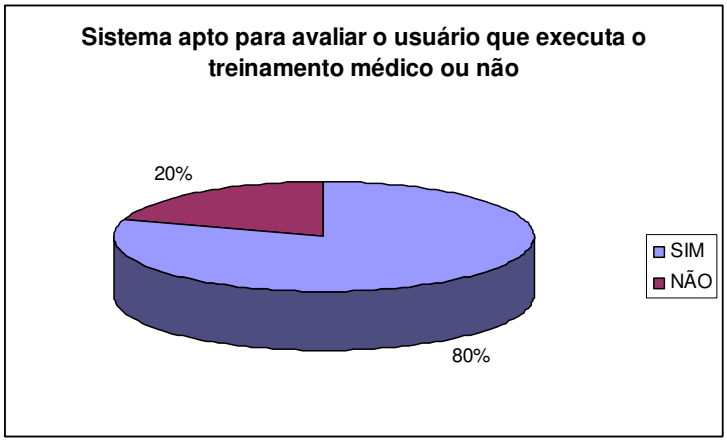


Figura 45: Gráfico que representa a aceitação da ferramenta.

Dentre todos os usuários que executaram o treinamento, foi unânime o comentário sobre a dificuldade de manipular a seringa utilizada no exame com a interação do mouse, concluindo-se a urgência em adicionar um dispositivo háptico à aplicação.

Na Figura 46 e na Figura 47 são apresentados relatórios de dois usuários que executaram o treinamento. O usuário 4 executou dois treinamentos. Apesar de no segundo treinamento o tempo gasto ter sido maior, sua nota aumentou significativamente visto que o usuário estava mais familiarizado ao ambiente e à manipulação da agulha, conseguindo encontrar o nódulo com apenas uma colisão na mama e um caminho não muito maior que o ideal.

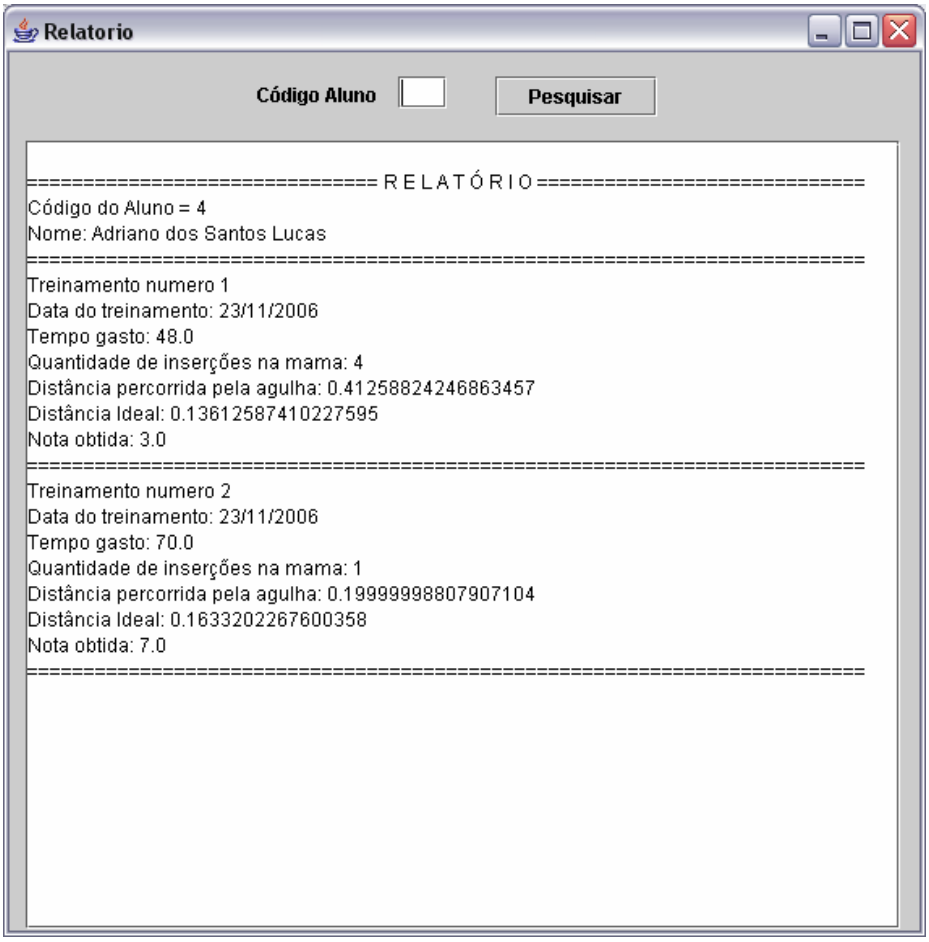


Figura 46: Relatório do treinamento do usuário 4.

O usuário 5 executou três treinamentos. Nota-se uma melhora significativa na nota, em conseqüência da diminuição do tempo, do número de inserções e da diferença entre as distâncias. No primeiro treinamento, este tempo é de trinta e quatro segundos e este número cai para dez no segundo e depois para cinco segundos no terceiro treinamento. O mesmo ocorre com o número de inserções da agulha na mama. Inicialmente, o usuário consegue atingir o nódulo depois de três inserções. Na segunda execução do treinamento este número reduz para duas inserções e no terceiro e último treinamento o usuário consegue atingir o nódulo com apenas uma inserção, que é o número ideal. A diferença entre as distâncias começa muito grande no primeiro treinamento e no terceiro essa diferença é mínima.

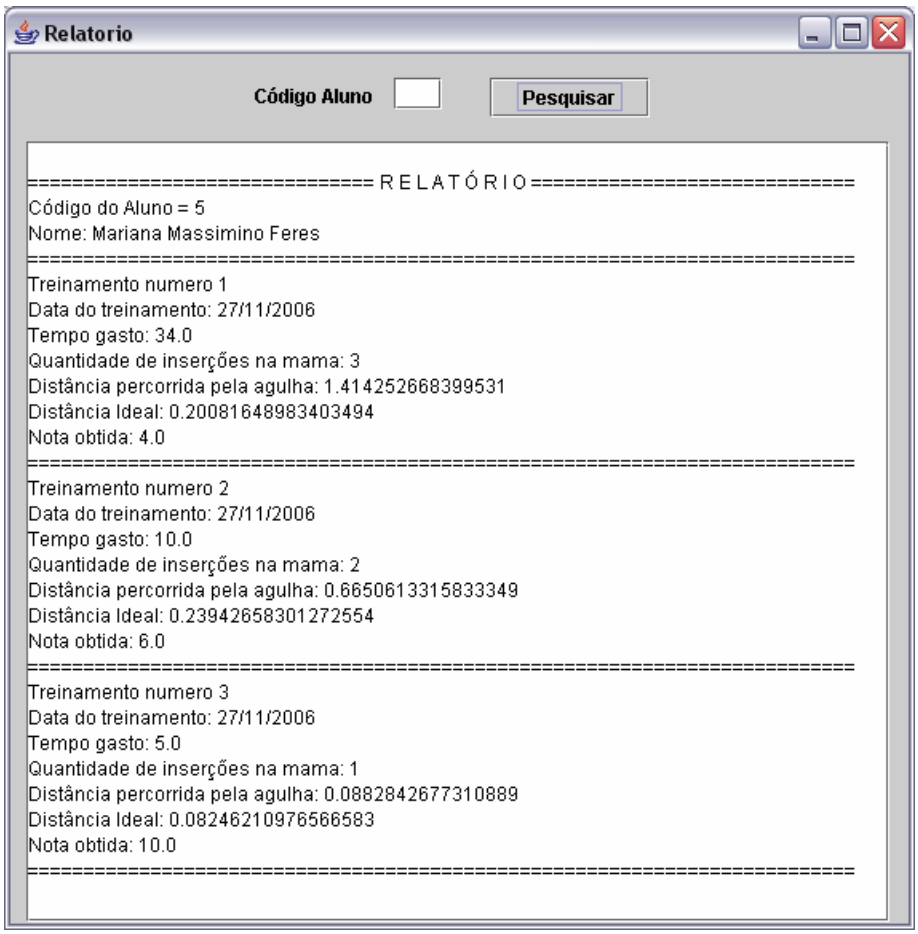


Figura 47: Relatório do treinamento do usuário 5.

3.2 Considerações Finais

Neste capítulo foram apresentados os resultados dos testes, executados por estudantes de Computação. Por meio dos resultados foi possível observar uma boa aceitação do módulo inserido no protótipo de simulação de exame de punção de mama, porém, ainda há necessidade de melhorias, como inserção de dispositivo háptico, para que com ele seja possível obter novos parâmetros a serem utilizados na avaliação da aprendizagem do usuário que executa o sistema.

CAPÍTULO 4 – CONCLUSÕES E TRABALHOS FUTUROS

A avaliação do usuário em sistemas de RV para treinamento ainda não é muito divulgada na literatura. Porém é muito importante porque possibilita identificar se há melhora de desempenho do usuário que executa o treinamento. Neste trabalho foram apresentados alguns projetos que já implementaram módulos de avaliação e foi proposta e implementada uma forma de avaliação em um simulador de exame de punção de mama.

Na ferramenta de avaliação implementada ainda não há muitos parâmetros para avaliar o usuário. Em trabalhos futuros pretende-se por meio de pesquisas, obter mais parâmetros para uma avaliação mais adequada como o ângulo de inserção da seringa e com a disponibilidade do dispositivo háptico, a força aplicada na inserção. Apesar dos poucos parâmetros utilizados, os resultados obtidos com os testes executados por usuários envolvidos no desenvolvimento de ferramentas de treinamento médico, indicaram desempenho satisfatório. No entanto ainda são necessários outros testes, os quais serão executados por usuários da área médica.

Para um maior realismo durante a execução do treinamento, pretende-se acrescentar ao protótipo de simulação de exame de punção de mama, uma classe para detecção de colisão com precisão implementada por KERA (2005), implementar a construção dinâmica do nódulo presente na mama, de acordo com a observação das imagens mamográficas de cada paciente e também substituir o mouse por dispositivo háptico na interação com o simulador. A construção dinâmica do nódulo será de extrema importância para que o usuário não se habitue ao posicionamento fixo do nódulo.

Neste projeto os pesos para cada categoria que compõe a nota podem ser escolhidos pelo aluno ou pelo professor. Para que somente o professor possa escolher os pesos de cada categoria, seria necessário criar uma visão para o professor. O professor se cadastraria e teria

a opção de escolher os pesos. Assim os seus alunos que executarem o treinamento receberão as notas de acordo com os pesos escolhidos anteriormente pelo professor. Esta visão que já está implementada ficaria somente para o usuário, onde ele pode se cadastrar no sistema, executar treinamentos e visualizar o relatório contendo todo o seu histórico.

O módulo de avaliação implementado neste projeto avalia o aluno a partir de alguns parâmetros muito importantes como a precisão, o tempo e o menor caminho a ser feito para atingir o nódulo, atribuindo ao aluno uma nota. Por meio dos relatórios foi possível concluir que com vários treinamentos executados há uma melhora visível desta nota, o que ajuda estes alunos a obterem conhecimento para, posteriormente, quando o forem aplicar na prática, estarem mais seguros e saberem como é e quais são as etapas do procedimento. A avaliação é imprescindível em qualquer etapa educacional, pois só assim pode ser medido o quanto as tarefas executadas estão contribuindo para a fixação do aprendizado dos alunos.

REFERÊNCIAS BIBLIOGRÁFICAS

ABRÃO, I. C.; ABRÃO, M. A. V. L; RAYEL, F. **QuestComp: Ferramenta via Web para Fixação e Avaliação da Aprendizagem.** In: II Workshop de Educação em Computação e Informática do Estado de Minas Gerais, 2003, Poços de Caldas. II Workshop de Educação em Computação e Informática do Estado de Minas Gerais, 2003.

ALBERIO, M de V; OLIVEIRA, J. C. **ACOnTECe-Cardio: um Ambiente Colaborativo para TrEinamento em Cirurgia Cardíaca**. In: VIII Simposium on Virtual Reality (SVR 2006), 2006, Belém, PA. Proceedings do VIII Simposium on Virtual Reality (SVR 2006), 2006. p. 397-408.

APACHE Derby. Disponível em: <http://db.apache.org/derby/index.html>. Acesso em Novembro de 2006.

BRAGA, M. Realidade Virtual e Educação**. Revista de Biologia e Ciências da Terra**, v. 1, n. 1, 2001.

BURDEA, G.; PATOUNAKIS, G.; POPESCU, V.; WEISS, R.E. **Virtual Reality Training for the Diagnosis of Prostate Cancer.** In: Information Technology Applications in Biomedicine, Washington, DC, Estados Unidos, 1998, p. 6-13.

ECLIPSE. Disponível em: <http://www.eclipse.org/>. Acesso em Novembro de 2006

GORMAN, P.; KRUMMEL, T.; WEBSTER, R.; SMITH, M.; HUTCHENS, D. **A Prototype Haptic Lumbar Puncture Simulator.** In: Medicine Meets Virtual Reality 2000, IOS Press, 2000.

GREENLEAF, W. J. **Medical Applications of Virtual Reality**. Overview, 2004.

KERA, M. **Detecção de colisão utilizando hierarquias em ferramentas de realidade virtual para treinamento médico**. 2005. Monografia (Bacharelado em Ciência da Computação) – Centro Universitário Eurípides de Marília, Fundação de Ensino Eurípides Soares da Rocha, 2005.

KIRNER, C. **Apostila do Ciclo de Palestras de Realidade Virtual.** Atividade do Projeto AVVIC – CNPq (Protem – CC – fase III) – DC/UFSCAR. São Carlos: 1996, p.1-10.

LIMA, L. de. **Protótipo de Ferramenta de Realidade Virtual para Simulação do Exame de Punção da Mama**. 2004. Dissertação (Mestrado em Ciência da Computação) - Centro Universitário Eurípides de Marília, Fundação de Ensino Eurípides Soares da Rocha, Marília, 2004.

MACHADO, L. S.; ZUFFO, M. K.; MORAES, R. M.; LOPES, R. D. **Modelagem Tátil, Visualização Estereoscópica e Aspectos de Avaliação em um Simulador de Coleta de Medula Óssea.** In: Proceedings of IV Symposium on Virtual Reality (SVR). Florianópolis. SBC Brazilian Computer Society, 2001, p. 23-31.

MACHADO, L.S. **A Realidade Virtual no Modelamento e Simulação de Procedimentos Invasios em Oncologia Pediátrica: Um Estudo de Caso no Transplante de Medula Óssea.** 2003. Grau: Tese (Doutorado em Engenharia Elétrica) – Escola Politécnica da Universidade de São Paulo, São Paulo, 2003.

MACHADO, L. S.; CAMPOS, S. F.; CUNHA, I. L. L.; MORAES, R. M. **Cybermed: Realidade Virtual Para Ensino Médico**. In: III CONGRESSO LATINO-AMERICANO DE ENGENHARIA BIOMÉDICA, 2004, João Pessoa. Proceedings of the International Federation for Medical and Biological Engineering, p. 573-576.

MACLEOD, N. **Evaluating Training Effectiveness**. *CAT*, pp-18 to 21.Vol 8. Issue 1, 1997.
MOREIRA, G. S. F.; ROCHA, A. R.; CAMPOS, F.; RABELO, L.; PORTO, E. **Um ambiente virtual para discussão de casos de Cardiologia.** In: Anais do XIII Simpósio Brasileiro de Informática na Educação, 2002.

NETTO, A. V.; MACHADO, L. S.; OLIVEIRA, M. C. F. Realidade Virtual – Definições, Dispositivos e Aplicações. **Revista Eletrônica de Iniciação Científica – Publicação da Sociedade Brasileira de Computação**, v. 2, n. 1, 2002.

PASSERINO, LILIANA MARIA; GELLER, MARLISE; SILVEIRA, SIDNEI RENATO; TAROUCO, LIANE M. R. **Aprendizagem e Avaliação em um Ambiente de Realidade Virtual Cooperativo de Aprendizagem (Projeto ARCA).** In: V Congresso Internacional de Informática na Educação - RIBIE 2000. Viña del Mar, Chile: 4-6 de dezembro de 2000.

SATAVA, R. **Medicine 2001: The King is Dead**. In: Virtual Reality Conference1994, 1994.

SILVA, R. L. S.; RAMOS, A. C. B. **Avaliação do Aprendizado do Sistema de Treinamento de Pilotos**. In: InfoCEFET2004 – II Simpósio de Informática do CEFET-PI, 2004.

SOUZA, D. F. L.; VALDEK, M. C. O.; MORAES, R. M.; MACHADO, L. S. **SITEG – Sistema Interativo de Treinamento em Exame Ginecológico**. In: Proceedings of VIII Symposium on Virtual Reality (SVR), Belém – PR, 2006.

STOTZKA, R.; MULLER, T.; EPPLER, W.; GEMMEKE, H. **Three-Dimensional Reconstruction of Clustered Microcalcifications from Two Digitized Mammograms.** SPIE, v. 3338, p.513-520, 1998.

SZÉKELY, Y. **Virtual Reality Based Surgery Simulation for Endoscopic Gynaecology, Studies in Health Technology and Informatics**, v. 62, p. 351-357, 705 Press, 1999.

VIRTUAL REALITIES, **Global Distributor of Quality Virtual Reality Products.** Disponível em: < http://www.vrealities.com> . Acesso em: Novembro de 2006.

WEBSTER, R W.; ZIMMERMAN, D. I.; MOHLER, B. J.; MELKONIAN, M. G.; HALUCK, R. S. **A Prototype Haptic Suturing Simulator.** In: Medicine Meets Virtual Reality 2001, IOS Press, 2001.

APÊNDICE A – Questionário

Dados Pessoais do Avaliador:

1 – Idade:
[ ] Abaixo de 20 anos
[ ] De 20 a 30 anos
[ ] De 31 a 40 anos
[ ] De 41 a 50 anos
[ ] De 51 a 60 anos
[ ] Acima de 60 anos

2 – Escolaridade:
[ ] Cursando Graduação
[ ] Graduação Completa
[ ] Especialização Completa
[ ] Cursando Mestrado
[ ] Mestrado Completo

3 – Profissão:
[ ] Estudante
[ ] Professor/Pesquisador
[ ] Profissional da Saúde
[ ] Outros

4 – Área de Atuação:
[ ] Engenharia
[ ] Computação
[ ] Saúde
[ ] Física
[ ] Outros

5 – Trabalha com Sistemas de Auxílio a Diagnóstico?
[ ] Não
[ ] Sim, com desenvolvimento
[ ] Sim, como usuário
[ ] Sim, como avaliador

Avaliação do Sistema:

1 – Facilidade de entendimento do sistema:
[ ] Ótimo
[ ] Bom
[ ] Regular
[ ] Ruim

2 – Facilidade de acesso às funções do sistema:
[ ] Ótimo
[ ] Bom
[ ] Regular
[ ] Ruim

3 – Facilidade de interpretação dos resultados exibidos no Relatório:
[ ] Ótimo
[ ] Bom
[ ] Regular
[ ] Ruim

4 – Considera este sistema apto para avaliar o usuário que executa o treinamento médico?
[ ] Sim
[ ] Não

5 – Comentários e sugestões

APÊNDICE B – Classe Aluno.java

```
package avaliausuario;

import java.sql.Connection;
import java.sql.SQLException;
import java.sql.ResultSet;
import javax.swing.*;

public class Aluno
{
   private int codAluno;
   private String nomeAluno;
   private int semestre;
   private String senhaUsuario;

   // construtor
   public Aluno( int cod, String nome, int sem, String senha)
   {
      codAluno = cod;
      nomeAluno = nome;
      semestre = sem;
      senhaUsuario = senha;

   }// termina construtor

//método para configurar o codigo do usuario
   public void setCodAluno( int cod )
   {
      codAluno = cod; //armazena o codigo
   }// termina setCodAluno

   //método para recuperar o codigo do usuario
   public int getCodAluno()
   {
      return codAluno;
   }// termina o método getCodAluno

// método para configurar o nome do aluno
   public void setNomeAluno( String nome )
   {
      nomeAluno = nome; // armazena o nome
   }// termina o método setNomeALuno

// método para recuperar o nome do aluno
   public String getNomeAluno()
   {
      return nomeAluno;
   }// termina o método getNomeAluno

// método para configurar o semestre em que o aluno esta matriculado
   public void setSemestre( int sem )
   {
      semestre = sem; // armazena o semestre
   }// termina o método setSemestre

// método para recuperar o semestre em que o aluno esta matriculado
   public int getSemestre()
   {
      return semestre;
```

```
   }// termina o método getSemestre

// método para configurar a senha
   public void setSenha( String senha )
   {
      senhaUsuario = senha;
   }// terminao método setSenha

// método para recuperar a senha
   public String getSenha()
   {
      return senhaUsuario;
   }// termina o método getSenha


/* * * *   I N S E R T D B ( )   M É T O D O   * * * */

  private boolean insertDB() {
    try {
      String insertSQL = "INSERT INTO aluno(codAluno, nomeAluno,
semestre,senha)  VALUES " + "(" + codAluno + ",'" + nomeAluno + "'," +
semestre + ",'" +senhaUsuario +"')";
     ConnectionManager.stmt.executeUpdate(insertSQL);
     ConnectionManager.con.commit();
    JOptionPane.showMessageDialog(null,"Usuário Cadastrado Com Sucesso");
      return true;
    }
    catch(SQLException e) {
      try {
        if (e.getErrorCode() == -193) {
          JOptionPane.showMessageDialog(null,"Já existe este usuário
cadastrado!" + e.getErrorCode() );
         return false;
        }
      }
      catch(Exception ex) {
        JOptionPane.showMessageDialog(null,"Problemas com Tratamento de
Erros!" );
       return true;
      }
    }
    return true;
  }

/* * * *   U P D A T E D B ( )   M É T O D O   * * * */

 private boolean updateDB() {
    try {
      String updateSQL = "UPDATE aluno SET nomeAluno = " + "'" + nomeAluno
+ "', semestre = "+ semestre + ", senha = '" + senhaUsuario + "' WHERE
codAluno = " + codAluno;
       ConnectionManager.stmt.executeUpdate(updateSQL);
      ConnectionManager.con.commit();
     JOptionPane.showMessageDialog(null,"Usuário Atualizado Corretamente");
       return true;
    }
    catch(SQLException e) {
      JOptionPane.showMessageDialog(null,"Problemas na Atualização do
Registro" + e.getErrorCode() );
      return false;
    }
```

```
  }

/* * * *    D E L E T E D B ( )    M É T O D O   * * * */

  private boolean deleteDB() {
    try {
      String deleteSQL = "DELETE FROM aluno WHERE codAluno = " + codAluno;
      ConnectionManager.stmt.executeUpdate(deleteSQL);
      ConnectionManager.con.commit();
     JOptionPane.showMessageDialog(null,"Usuário Removido Com Sucesso");
      return true;
    }
    catch(SQLException e) {
      JOptionPane.showMessageDialog(null,"Problemas na Remoção do Registro"
+ e.getErrorCode() );
      return false;
    }
  }

//* * * *    F I N D L I K E D B ( )    M É T O D O   * * * *

  private ResultSet findlikeDB() {
    try {
      String findlikeSQL = "SELECT * FROM aluno WHERE codAluno = "+
codAluno;

      ResultSet rs = ConnectionManager.stmt.executeQuery(findlikeSQL);
      return rs;
    }
    catch(SQLException e) {
      JOptionPane.showMessageDialog(null,"Problemas na Consulta do Usuário"
+ e.getErrorCode() );
     return null;
    }
  }

// Metodos públicos que são intermediários aos métodos que cuidam da
manipulação no banco de dados

/* * * *    S A V E ( )    M É T O D O   * * * */

 public boolean save(){
    ResultSet rs;
    try{
      rs = findlikeDB();
      // verifica se o usuário será atualizado ou inserido no DB
      if (rs.next()){
        return updateDB();
      }
      else{
        return insertDB();
      }
    }
    catch(SQLException e){
      JOptionPane.showMessageDialog(null,"Erro: " + e.getErrorCode() +
"msg:"+ e.getMessage());
     return false;
    }
  }

/* * * *    D E L E T E ( )    M É T O D O   * * * */
```

```
public boolean delete(){
return deleteDB();
}

/* * * *    F I N D L I K E ( )    M É T O D O   * * * */

public ResultSet findlike(){
    return findlikeDB();
}

/* * * *    G E T L A S T ( )    M É T O D O   * * * */
/* * * * Pega o valor do código do último usuário *  */
public static int getLast()
   {
      ConnectionManager.ConnectDB("bdavaliacao");
      int ultimo = 0;
      try
      {
         String consultar = "SELECT max(codAluno) as maximo FROM aluno";
         java.sql.ResultSet rs =
ConnectionManager.stmt.executeQuery(consultar);

         if (rs.next())
         {
            ultimo = rs.getInt("maximo");
         }

         ConnectionManager.CloseDB();
      }
      catch (java.sql.SQLException e)
      {
         e.getMessage();
      }
      return ultimo;
   }// fim getLast

}// fim da classe Aluno
```

APÊNDICE C – Classe ConnectionManager.java

```
package avaliausuario;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.Statement;
import java.sql.SQLException;

public class ConnectionManager    //definindo classe de conexão!!
{
    public static Connection con; // atributo....inserir , atualizar
    public static Statement stmt; //atributo ....manipular dados no banco

   // Conexao - carrega o driver e realiza a conexao com o banco
        public static void ConnectDB (String fonte) //
        {
         try //tratamento de excessões , se der erro vai para o catch
         {
          String url = "jdbc:derby:" + fonte;
          Class.forName( "org.apache.derby.jdbc.EmbeddedDriver");
          con = DriverManager.getConnection(url);
          stmt = con.createStatement();
          con.setAutoCommit(true);
          }

    catch (Exception e)
    {
     e.printStackTrace();
    }
        }
  // encerra o statement e a conexao

      public static void CloseDB()
      {
   try
   {
       stmt.close(); // encerra objeto statment
       con.close(); // encerra objeto connection
   }

       catch (Exception e)
       {
          e.printStackTrace();
       }
      }
} // fecha classe ConnectionManager
```

APÊNDICE D – Classe TelaManutencao.java

```
package avaliausuario;

import java.awt.*;
import javax.swing.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.lang.Exception;
import javax.swing.JOptionPane;

public class TelaManutencao extends JFrame {
   private JLabel codigoUsuario, nomeUsuario, semUsuario, senhaUsuario;
   private JTextField codCampo, nomeCampo, semCampo;
   private JPasswordField senhaCampo;

    public TelaManutencao() {

        setTitle("Usuário");
      setSize( 570, 260 );
      setLocation(215, 270);

      Container container = getContentPane();
        container.setLayout(null);

      /* Botões */
      JButton cadastrarUsuario = new JButton(" Cadastrar ");
      container.add(cadastrarUsuario);
      cadastrarUsuario.setBounds(new Rectangle(2,180, 187, 46));

      JButton removerUsuario = new JButton(" Remover ");
      container.add(removerUsuario);
      removerUsuario.setBounds(new Rectangle(375,180, 187, 46));

      JButton atualizarUsuario = new JButton(" Atualizar ");
      container.add(atualizarUsuario);
      atualizarUsuario.setBounds(new Rectangle(188,180, 187, 46));

      /* Labels */
      codigoUsuario = new JLabel(" Código ");
      container.add(codigoUsuario);
      codigoUsuario.setBounds(new Rectangle(30, 18, 94, 16));

      nomeUsuario = new JLabel(" Nome ");
      container.add(nomeUsuario);
      nomeUsuario.setBounds(new Rectangle(30, 62, 94, 16));

      semUsuario = new JLabel(" Semestre");
      container.add(semUsuario);
      semUsuario.setBounds(new Rectangle(30, 102, 94, 16));

      senhaUsuario = new JLabel(" Senha ");
      container.add(senhaUsuario);
      senhaUsuario.setBounds(new Rectangle(30, 140, 94, 16));

      /* TextFields */
```

```
        codCampo = new JTextField();
        container.add(codCampo);
        codCampo.setBounds(new Rectangle(150, 15, 30, 20));
        codCampo.requestFocus();
        codCampo.setText(String.valueOf(Aluno.getLast()+1));

        nomeCampo = new JTextField();
        container.add(nomeCampo);
        nomeCampo.setBounds(new Rectangle(150, 58, 350, 20));

        semCampo = new JTextField();
         container.add(semCampo);
        semCampo.setBounds(new Rectangle(150, 102, 30, 20));

        senhaCampo = new JPasswordField();
        container.add(senhaCampo);
        senhaCampo.setBounds(new Rectangle(150, 140, 100, 20));

        /* ACTIONS */
        /* BUTTON HANDLER */
// para tratamento de evento dos JButtons
ButtonHandler handler = new ButtonHandler();
        cadastrarUsuario.addActionListener( handler );
        removerUsuario.addActionListener( handler );
        atualizarUsuario.addActionListener( handler );

    }
             private class ButtonHandler implements ActionListener{

              public void actionPerformed ( ActionEvent event )
                {
                 ConnectionManager.ConnectDB("bdavaliacao"); //conecta


                 //lê os textfields e armazena
                 {
                    int codigo = Integer.parseInt(codCampo.getText());
                    String nome = nomeCampo.getText();
                    int sem = Integer.parseInt(semCampo.getText());
                    String senha = senhaCampo.getText();

                    // cria um objeto aluno
                    Aluno user = new Aluno(codigo,nome, sem,senha);

                    if (event.getActionCommand()== " Cadastrar ")
                        {
                           user.save(); // testa, se jah existe, atualiza,
senão insere!!
                        }

                    else if(event.getActionCommand() == " Atualizar ")
                        {
                           user.save();
                        }
                    else user.delete();//voltar como estava, nao atualiza

                 ConnectionManager.CloseDB(); //encerra conexão

                 // limpar os campos na tela
                 codCampo.setText(String.valueOf(Aluno.getLast()+1));
```

```
                nomeCampo.setText("");
                semCampo.setText("");
                senhaCampo.setText("");

            } // end method actionPerformed
        } // end anonymous inner class
}
}
```

APÊNDICE E – Classe TelaRelatorio.java

```
package avaliausuario;

import gui.MenuPuncao;
import java.awt.*;
import javax.swing.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.ResultSet;
import java.sql.ResultSetMetaData;
import java.sql.SQLException;
import java.lang.Exception;
import javax.swing.JOptionPane;

public class TelaRelatorio extends JFrame {

    private JTextArea areaSelect;
    private JLabel codigoUsuario, tempoIdeal,
    pesoTempo,pesoDistancia,pesoInsercao;
    private JButton pesquisaUsuario;
    private JTextField codCampo,tempoICampo,
    tempoCampo,distCampo,inserCampo;
    private String nomeAluno;
    private int semestre, numeroT,numInsercao,cont;
    private double NotaFinal,notas, media,distIdeal,distancia,tempo;

     public TelaRelatorio() {

        setTitle("Relatorio");
        setSize( 570, 600 );
        setLocation(215, 100);

        Container container = getContentPane();
        container.setLayout(null);

        /*Área de Texto */
        areaSelect = new JTextArea();
        JScrollPane scroll = new JScrollPane(areaSelect);
        container.add(scroll);
        scroll.setBounds( new Rectangle(10,70,540,490));

        /* Botão */
        pesquisaUsuario = new JButton(" Avaliar ");
        container.add(pesquisaUsuario);
        pesquisaUsuario.setBounds(new Rectangle(300,15, 100, 25));

        /* Rótulos */
        codigoUsuario = new JLabel(" Código Aluno ");
        container.add(codigoUsuario);
        codigoUsuario.setBounds(new Rectangle(150, 18, 94, 16));

        tempoIdeal = new JLabel ("Tempo Ideal");
        container.add(tempoIdeal);
        tempoIdeal.setBounds(new Rectangle(20, 45, 94, 16));

        pesoTempo = new JLabel("Peso Tempo");
        container.add(pesoTempo);
```

```
        pesoTempo.setBounds(new Rectangle(140, 45, 94, 16));

        pesoDistancia = new JLabel("Peso Distância");
        container.add(pesoDistancia);
        pesoDistancia.setBounds(new Rectangle(280, 45, 94, 16));

        pesoInsercao = new JLabel("Peso Inserção");
        container.add(pesoInsercao);
        pesoInsercao.setBounds(new Rectangle(420, 45, 94, 16));

        /* Campos */
        codCampo = new JTextField();
        container.add(codCampo);
        codCampo.setBounds(new Rectangle(240, 15, 30, 20));
        codCampo.requestFocus();

        tempoICampo = new JTextField();
        container.add(tempoICampo);
        tempoICampo.setBounds( new Rectangle(100, 45, 30, 20));

        tempoCampo = new JTextField();
        container.add(tempoCampo);
        tempoCampo.setBounds( new Rectangle(220, 45, 30, 20));

        distCampo = new JTextField();
        container.add(distCampo);
        distCampo.setBounds( new Rectangle(370, 45, 30, 20));

        inserCampo = new JTextField();
        container.add(inserCampo);
        inserCampo.setBounds( new Rectangle(510, 45, 30, 20));

        /* Action Botão A V A L I A R */
        pesquisaUsuario.addActionListener(
               new ActionListener() // anonymous inner class
               {
                  public void actionPerformed( ActionEvent event )
                  {
                     int codAluno   = Integer.parseInt(codCampo.getText());


                        ConnectionManager.ConnectDB("bdavaliacao");

                        try{
                           String consultaAluno = "SELECT nomeAluno,semestre
from aluno where codAluno = "+codAluno;
                           java.sql.ResultSet rs1 =
ConnectionManager.stmt.executeQuery(consultaAluno);

                           if(rs1.next())
                           {
                                nomeAluno = rs1.getString("nomeAluno");
                                semestre = rs1.getInt("semestre");

                              try
                              {
                                 String consultar = "SELECT * from
treinamento where codAluno = "+ codAluno;
                                 java.sql.ResultSet rs =
ConnectionManager.stmt.executeQuery(consultar);
```

```
                                    if(rs.next())
                                    {       media = 0;
                                          cont = 0;
                                          areaSelect.setText("");

   areaSelect.append("\n================================ R E L A T Ó R I O
==============================");
                                          areaSelect.append("\nCódigo do
Aluno = "+codAluno);
                                          areaSelect.append("\nNome:
"+nomeAluno);
                                          areaSelect.append("\nSemestre em
que está matriculado: "+semestre);

   areaSelect.append("\n================================================
=========================");
                                          int codTreinamento =
rs.getInt("codTreinamento");
                                          areaSelect.append("\nTreinamento
numero "+codTreinamento);

                                          String data = rs.getString("data");
                                          areaSelect.append("\nData do
treinamento: "+data);

                                           tempo = rs.getDouble("tempo");
                                          areaSelect.append("\nTempo gasto:
"+tempo);

                                          numInsercao =
rs.getInt("numInsercoes");
                                          areaSelect.append("\nQuantidade de
inserções na mama: "+numInsercao);

                                           distancia =
rs.getDouble("distancia");
                                          areaSelect.append("\nDistância
percorrida pela agulha: "+distancia);

                                          distIdeal =
rs.getDouble("distIdeal");
                                          areaSelect.append("\nDistância
Ideal: "+distIdeal);


                                           double Nota = avalia();
                                           media += Nota;
                                           cont++;
                                           areaSelect.append("\nNota Final:
"+ Nota);
                                          /*double NotaFinal =
rs.getDouble("NotaFinal");
                                          areaSelect.append("\nNota obtida:
"+ NotaFinal);*/

   areaSelect.append("\n================================================
=========================");
                                          codCampo.setText("");

                                          while (rs.next())
                                          {
```

```
                                        codTreinamento =
rs.getInt("codTreinamento");

areaSelect.append("\nTreinamento numero "+codTreinamento);

                                        data = rs.getString("data");
                                        areaSelect.append("\nData do
treinamento: "+data);

                                        tempo = rs.getDouble("tempo");
                                        areaSelect.append("\nTempo
gasto: "+tempo);

                                        numInsercao =
rs.getInt("numInsercoes");
                                        areaSelect.append("\nQuantidade
de inserções na mama: "+numInsercao);

                                        distancia =
rs.getDouble("distancia");
                                        areaSelect.append("\nDistância
percorrida pela agulha: "+distancia);

                                        distIdeal =
rs.getDouble("distIdeal");
                                        areaSelect.append("\nDistância
Ideal: "+distIdeal);

                                        double Notas = avalia();
                                        areaSelect.append("\nNota
Final: "+ Notas);

                                        media += Notas;
                                        cont++;


areaSelect.append("\n=================================================
======================");
                                        codCampo.setText("");

                                    }
                                    areaSelect.append("\nMEDIA: "+
media/cont);
                                }
                                else
                                {
                                    areaSelect.setText("");

   JOptionPane.showMessageDialog(null,"Usuário não executou nenhum
treinamento!!");
                                    codCampo.setText("");


                                }
                            }
                            catch (java.sql.SQLException e)
                            {
                                e.getMessage();
                            }

                        }
```

```
                           else
                           {
                              areaSelect.setText("");
                              JOptionPane.showMessageDialog(null,"Usuário
não cadastrado!!");
                               codCampo.setText("");

                           }
                        }
                        catch (java.sql.SQLException e)
                        {
                           e.getMessage();
                        }



                     ConnectionManager.CloseDB();
                  } // end actionPerformed
               }
      ); // end addActionListener

   }

    /* ------------------------------------------------------------------
------------------*/
    /* Calcula a nota  do usuário no treinamento de acordo com os dados
obtidos no treinamento*/
    public double avalia()
    {
       int numIdeal = 1;
       double NotaD,NotaI,NotaT,tempo1,dist1;

       int tempoIdeal = Integer.parseInt(tempoICampo.getText());
       int pesoTempo  = Integer.parseInt(tempoCampo.getText());
       int pesoDist   = Integer.parseInt(distCampo.getText());
       int pesoInser  = Integer.parseInt(inserCampo.getText());
          dist1 = distancia < distIdeal ? distIdeal : distancia;
          NotaD = (distIdeal/dist1) * pesoDist;
          NotaI = ((double)numIdeal/numInsercao) * pesoInser;
          tempo1 = tempo < tempoIdeal ? tempoIdeal :tempo;
          NotaT = (double)(tempoIdeal/tempo1) * pesoTempo;
          NotaFinal = (double)Math.round( NotaD + NotaI + NotaT);
          return NotaFinal;
    }
    /* ------------------------------------------------------------------
----------------*/
}
```

APÊNDICE F – Classe MenuPrincipal.java

```
package avaliausuario;

import javax.swing.*;
import java.awt.event.*;
import java.awt.*;
import gui.*;

public class MenuPrincipal extends JFrame {

   private GridBagLayout layout,layoutR;
   private GridBagConstraints constraints,constraintsR;
   private Container container;
   private   JPanel painelLogin;
   private JPanel painelRelatorio;
   private JPasswordField campoSenha;
   private JTextField campoCodigo,codigo;

  public MenuPrincipal() {

      setTitle("Exame de Punção de Mama - Avaliação do Usuário");
      setLocation(100,100);

      // Cria uma barra de menu para o JFrame
      JMenuBar menuBar = new JMenuBar();

      // Adiciona a barra de menu ao frame
      setJMenuBar(menuBar);

      // Define e adiciona menu drop down na barra de menus
      JMenu fileMenu = new JMenu("Arquivo");
      menuBar.add(fileMenu);
     fileMenu.setMnemonic( 'A' ); // set mnemonic to A

      /* Cria e adiciona itens simples para o menu */

      /* MANUTENÇÃO */
       JMenuItem manutencaoAction = new JMenuItem("Manutenção de
Usuários");
       manutencaoAction.setMnemonic( 'M' ); // set mnemonic to M
       fileMenu.add(manutencaoAction);
       manutencaoAction.addActionListener(

       new ActionListener()
       {
          public void actionPerformed( ActionEvent event )
          {
             TelaManutencao tela = new TelaManutencao();
          //  tela.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
          tela.setVisible( true ); // mostra o frame

          } //end método ActionPerfored
       }
    ); //fim addActionListener

       /* S A I R */
       JMenuItem exitAction = new JMenuItem("Sair");
```

```
    exitAction.setMnemonic( 'S' ); // set mnemonic to S
    fileMenu.add(exitAction);
    exitAction.addActionListener(
           new ActionListener()
           {

              public void actionPerformed( ActionEvent event )
              {
                  System.exit( 0 ); // fecha aplicação
              } // end actionPerformed
           }
    ); // end addActionListener

    JMenu exibeMenu = new JMenu("Exibir");
    menuBar.add(exibeMenu);
    exibeMenu.setMnemonic( 'E' ); // set mnemonic to E

    /* R E L A T Ó R I O */
     JMenuItem relatAction = new JMenuItem("Relatório");
    relatAction.setMnemonic( 'R' ); // set mnemonic to S
    exibeMenu.add(relatAction);
    relatAction.addActionListener(
       new ActionListener()
       {
           public void actionPerformed( ActionEvent event)
           {
             TelaRelatorio relatorio = new TelaRelatorio();
             relatorio.setVisible( true ); // mostra o frame
           }
        }
    );

    /* L O G I N */
     container = getContentPane();
    layout = new GridBagLayout();
    //layoutR = new GridBagLayout();
    container.setLayout(new BorderLayout());

    painelLogin = new JPanel(layout);

    constraints = new GridBagConstraints();
    container.add(painelLogin,BorderLayout.CENTER);

    JLabel labelUsuario = new JLabel("Código do Usuário");
    JLabel labelSenha = new JLabel("Senha");

    campoCodigo = new JTextField(10);
    campoSenha = new JPasswordField();

    JButton botaoEntrar = new JButton("Entrar");

    constraints.fill = GridBagConstraints.HORIZONTAL;
    addComponent(labelUsuario,0,0,1,1);
    addComponent(labelSenha,1,0,1,1);
    addComponent(campoCodigo,0,1,1,1);
    addComponent(campoSenha,1,1,1,1);
    addComponent(botaoEntrar,2,1,1,1);

/* Action Botão E N T R A R */
    botaoEntrar.addActionListener(
           new ActionListener() // anonymous inner class
```

```
            {
                public void actionPerformed( ActionEvent event )
                {
                    int codAluno = Integer.parseInt(campoCodigo.getText());
                    String senha = campoSenha.getText();
                    ConnectionManager.ConnectDB("bdavaliacao");
                        try
                        {
                            String consultar = "SELECT * from aluno where
codAluno = "+ codAluno + " and senha = '"+ senha +"'";
                            java.sql.ResultSet rs =
ConnectionManager.stmt.executeQuery(consultar);

                            if (rs.next())
                            {
                                MenuPuncao principal = new
MenuPuncao(codAluno);
                            }
                            else
                                JOptionPane.showMessageDialog(null,"Usuário
e/ou senha inválidos!!");
                                campoCodigo.setText("");
                                campoSenha.setText("");
                        }

                        catch (java.sql.SQLException e)
                        {
                            e.getMessage();
                        }
                    ConnectionManager.CloseDB();
                } // end actionPerformed
            }
     ); // end addActionListener
    }

private void addComponent( Component component, int row, int column, int
width, int height )
    {

        constraints.gridx = column;
        constraints.gridy = row;

        constraints.gridwidth = width;
        constraints.gridheight = height;

        layout.setConstraints( component, constraints );
        painelLogin.add(component);

    }

    /* * * *  M A I N  * * * */

    public static void main( String args[] )
        {
            MenuPrincipal menuUsuario = new MenuPrincipal(); // cria
MenuUsuario
            menuUsuario.setSize( 800, 600 ); // seta tamanho do frame
            menuUsuario.setVisible( true ); // mostra o frame
        } // end main

}// end MenuPrincipal
```

APÊNDICE G – Classe ColisaoNodulo.java

```
package gui;

import java.util.Enumeration;
import javax.media.j3d.Behavior;
import javax.media.j3d.Shape3D;
import javax.media.j3d.WakeupOnCollisionEntry;
import javax.media.j3d.WakeupOnCollisionExit;
import javax.media.j3d.WakeupOnCollisionMovement;
import javax.media.j3d.WakeupCriterion;
import javax.media.j3d.*;
import javax.vecmath.*;

import avaliausuario.*;

/* Colisão para o nódulo */

public class ColisaoNodulo extends Behavior {

    public boolean inCollision = false;
    private Shape3D shape;
    public float x,y,z;
    public int codTreinamento,codAluno;
    public long tempofim;

    private WakeupOnCollisionEntry wEnter;
    private WakeupOnCollisionExit wExit;
    private WakeupOnCollisionMovement wMovement;

    /* Construtor */
    public ColisaoNodulo(Shape3D s,int cod, int treina) {
        shape = s;
        inCollision = false;
        codAluno = cod;
        codTreinamento = treina;
    }

    public void initialize()
    {
        wEnter = new WakeupOnCollisionEntry(shape);
        wExit = new WakeupOnCollisionExit(shape);
        wMovement = new WakeupOnCollisionMovement(shape);

        WakeupCriterion vetor[] = new WakeupCriterion[3];
        vetor[0] = wMovement;
        vetor[1] = wEnter;
        vetor[2] = wExit;

        wakeupOn(wEnter);
    }

    public void processStimulus(Enumeration criteria)
    {
        while (criteria.hasMoreElements())
        {
            Object o = criteria.nextElement();
```

```
            if (o==wEnter)
            {
               inCollision = true;
               tempofim = System.currentTimeMillis();
               Vector3f v3fnodulo = new Vector3f();
               SceneGraphPath sg1 =wEnter.getTriggeringPath();//Objeto que
causou a colisão
               Transform3D t3d1 = new Transform3D();
               sg1.getObject().getLocalToVworld(t3d1);
               t3d1.get(v3fnodulo);

               x = v3fnodulo.x;
               y = v3fnodulo.y;
               z = v3fnodulo.z;
               wakeupOn(wExit);
            }
            else if(o==wExit)
            {
               inCollision = false;
               wakeupOn(wEnter);
            }
      }
   }
}
```

APÊNDICE H – Classe ColisaoMama.java

```
package gui;

import avaliausuario.*;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.Enumeration;

import javax.media.j3d.Behavior;
import javax.media.j3d.Shape3D;
import javax.media.j3d.WakeupOnCollisionEntry;
import javax.media.j3d.WakeupOnCollisionExit;
import javax.media.j3d.WakeupOnCollisionMovement;
import javax.media.j3d.WakeupOr;
import javax.media.j3d.WakeupCriterion;
import javax.media.j3d.*;
import javax.swing.JOptionPane;
import javax.vecmath.*;
import java.lang.Math;
import java.util.Date;

/* Colisão para a mama */
public class ColisaoMama extends Behavior {

   public boolean inCollision = false;
   private Shape3D shape;
   public int numInsercao=0,numMovimento=0;
   public float x,y,z,a,b,c;
   public int codTreinamento,codAluno;
   public long tempoinicio;
   public double tempo;
   public double distancia,distIdeal,dist,NotaFinal;

   Date data = new Date();
   java.text.DateFormat sdf =new java.text.SimpleDateFormat("dd/MM/yyyy");
   String dt = sdf.format(data);

   private WakeupOnCollisionEntry wEnter;
   private WakeupOnCollisionExit wExit;
   private WakeupOnCollisionMovement wMovement;
   private WakeupOr ou;
   private ColisaoNodulo cn;

   private Exame e;

   /* Construtor */
   public ColisaoMama( Shape3D s, int cod, int treina, ColisaoNodulo
nodulo, Exame e){
       shape = s;
       codAluno = cod;
       codTreinamento = treina;
       cn = nodulo;
       inCollision = false;
       this.e = e;
   }

   public void initialize()
```

```
    {
        wEnter = new WakeupOnCollisionEntry(shape);
        wExit = new WakeupOnCollisionExit(shape);
        wMovement = new WakeupOnCollisionMovement(shape);
        WakeupCriterion vetor[] = new WakeupCriterion[3];
        vetor[0] = wMovement;
        vetor[1] = wEnter;
        vetor[2] = wExit;
        ou = new WakeupOr(vetor);

        wakeupOn(wEnter);
        tempoinicio = System.currentTimeMillis();

    }

    public void processStimulus(Enumeration criteria)
    {
        ConnectionManager.ConnectDB("bdavaliacao");//conecta no banco

        while (criteria.hasMoreElements())
        {
            Object o = criteria.nextElement();

                if (o==wEnter) //se colidiu
                {
                    inCollision = true;
                    numInsercao = numInsercao + 1;
                    wakeupOn(ou); // acorda com qualquer evento
                }

                else if(o==wMovement) //se movimentou
                {
                    if(!cn.inCollision)// se o nódulo não está em colisão com a
agulha
                    {
                        Vector3f v3fmama = new Vector3f();
                        SceneGraphPath scenemama =
wMovement.getTriggeringPath(); //Objeto que causou a colisão

                        Transform3D transfmama = new Transform3D();
                        scenemama.getObject().getLocalToVworld(transfmama);
                        transfmama.get(v3fmama);//armazena as coordenadas de
cada movimentacao no Vector3f

                        x = v3fmama.x;// obtém-se os valores da coordenada
separadamente em x
                        y = v3fmama.y;// y
                        z = v3fmama.z;// e z

                        numMovimento ++; // incrementa o código da coordenada
                        //chamada do método que insere as coordenadas na tabela
coordenada
                        insereCoord(); // insere a coordenada no banco
                        wakeupOn(ou); //acorda com qualquer evento
                    }
                    else
                    {
                        tempo = (cn.tempofim - tempoinicio)/1000; // calcula o
tempo gasto no exame
                        a = cn.x; //obtém-se os valores da coordenada
separadamente em a
```

```
                    b = cn.y; //b
                    c = cn.z; //c
                    x = cn.x;
                    y = cn.y;
                    z = cn.z;
                    numMovimento ++;// incrementa o código da coordenada
                    //camada do método que insere as coordenadas na tabela
coordenada
                    insereCoord(); // insere a coordenada do nódulo
                    distIdeal();// calcula a distancia ideal
                    distancia();// calcula a distancia percorrida no exame
                    updateTreinamento();// atualiza a tabela treinamento
                    JOptionPane.showMessageDialog(null,"Nódulo em contato!
Exame terminado");
                    e.setVisible(false);

                }
            }

            else if(o==wExit) // se saiu da colisão
            {
                inCollision = false;
                wakeupOn(wEnter); // acorda com uma colisão
            }
        }
    }

    /*-------------------------------------------------------------------
-------------------------------------------------------------------
------------------------------*/
    /* Calcula a distância ideal entre a colisão da mama e do nódulo */
    public void distIdeal()
    {   float x,y,z;

        try
        {
            String consultar ="SELECT x,y,z FROM coordenada WHERE
codTreinamento = "+ codTreinamento + " and codCoordenada = " +1;
          ResultSet rs = ConnectionManager.stmt.executeQuery(consultar);
          if (rs.next())
          {
              x = rs.getFloat("x");
              y = rs.getFloat("y");
              z = rs.getFloat("z");

              distIdeal = Math.sqrt(Math.pow((x - a),2)+Math.pow((y -
b),2)+Math.pow((z - c),2));

          }

          else JOptionPane.showMessageDialog(null,"ERRO!!");
        }
        catch (java.sql.SQLException e)
        {
            e.printStackTrace();

        }

    }
```

```
   /*-------------------------------------------------------------------------
-------------------------------------------------------------------------
------------------------------*/
   /* Calcular a o caminho percorrido entre a colisão da mama e do nódulo
*/
   public void distancia()
   {
      int i,j,k;
      float vx[],vy[],vz[];
      double dist[];
      vx = new float[numMovimento];
      vy = new float[numMovimento];
      vz = new float[numMovimento];
      dist = new double [numMovimento];

      try
      {
          String consulta ="SELECT x,y,z FROM coordenada WHERE codAluno ="
+ codAluno + " and codTreinamento = "+ codTreinamento;
         ResultSet rs = ConnectionManager.stmt.executeQuery(consulta);
         i=0;
         while (rs.next())
            {
                vx[i] = rs.getFloat("x");
                vy[i] = rs.getFloat("y");
                vz[i] = rs.getFloat("z");
                i++;
            }
         }
      catch (java.sql.SQLException e)
      {
         e.printStackTrace();

      }
      j=0; k=1;
      do
      {
            dist[j] = Math.sqrt(Math.pow((vx[j] -
vx[k]),2)+Math.pow((vy[j] - vy[k]),2)+Math.pow((vz[j] - vz[k]),2));
            distancia = distancia + dist[j];
            j++;k++;


      }while(j != numMovimento-1);
   }

   /*-------------------------------------------------------------------------
-------------------------------------------------------------------------
------------------------------*/
   /* Insere as coordenadas de cada movimento da agulha na tabela
coordenada */
   public boolean insereCoord()
   {
          try {
                String insertSQL = "INSERT INTO
coordenada(codAluno,codTreinamento,codCoordenada,x,y,z)  VALUES " + "(" +
codAluno +","+ codTreinamento +","+ numMovimento + "," + x + "," + y + ","
+ z + ")";
                ConnectionManager.stmt.executeUpdate(insertSQL);
                ConnectionManager.con.commit();
                return true;
```

```
            }
           catch(SQLException e) {
                 try {
                    if (e.getErrorCode() == -193) {
                     return false;
                    }
                    else
                    System.out.println(e.getMessage());
                 }
                 catch(Exception ex) {
                    System.out.println("Problema com tratamento de
erros!");
                    return true;
                 }
           }
           return true;
    }
    /*-------------------------------------------------------------------------
-------------------------------------------------------------------------
------------------------------*/
    /* Atualiza a tabela treinamento depois de terminado */
    private boolean updateTreinamento() {
        try {
           String updateSQL = "UPDATE treinamento SET data = '"+ dt + "',
tempo = " + tempo + ",numInsercoes = "+numInsercao+", distancia = " +
distancia + ", distIdeal = "+ distIdeal + " WHERE codAluno = " + codAluno +
" and codTreinamento = " + codTreinamento;
           ConnectionManager.stmt.executeUpdate(updateSQL);
           ConnectionManager.con.commit();
           return true;
        }
        catch(SQLException e) {

           return false;
        }
      }

    /*-------------------------------------------------------------------------
-------------------------------------------------------------------------
------------------------------*/
}// end ColisaoMama
```

APÊNDICE I – Classe Exame.java

```
package gui;
import java.awt.GraphicsConfiguration;

import java.awt.*;
import java.awt.event.*;
import java.sql.SQLException;

import javax.media.j3d.AmbientLight;
import javax.media.j3d.Background;
import javax.media.j3d.BoundingSphere;
import javax.media.j3d.BranchGroup;
import javax.media.j3d.Canvas3D;
import javax.media.j3d.DirectionalLight;
import javax.media.j3d.Group;
import javax.media.j3d.Shape3D;
import javax.media.j3d.Transform3D;
import javax.media.j3d.TransformGroup;
import javax.swing.ImageIcon;
import javax.swing.JLabel;
import javax.vecmath.AxisAngle4d;
import javax.vecmath.Color3f;
import javax.vecmath.Point3d;
import javax.vecmath.Point3f;
import javax.vecmath.Vector3f;
import javax.media.j3d.Node;


import avaliausuario.ConnectionManager;

import com.sun.j3d.loaders.Scene;
import com.sun.j3d.loaders.objectfile.ObjectFile;
import com.sun.j3d.utils.behaviors.mouse.MouseRotate;
import com.sun.j3d.utils.behaviors.mouse.MouseTranslate;
import com.sun.j3d.utils.behaviors.mouse.MouseZoom;
import com.sun.j3d.utils.universe.SimpleUniverse;

public class Exame extends  javax.swing.JInternalFrame
{
    public Canvas3D canvas = null;
    public SimpleUniverse simpleU;
    public BranchGroup raizSeringa;
    public BranchGroup raizMama;
    public BranchGroup sceneRoot;
    public BranchGroup branch;

    public Point3f center;
    public Point3f centerem;

    public Vector3f vec;
    public Vector3f themove;

    public TransformGroup sceneTG;
    public TransformGroup objTG;
    public TransformGroup objTGEM;
    public TransformGroup tgembolo;
    public TransformGroup objTGAG;
    public TransformGroup tgagulha;
    public TransformGroup objTGTB;
```

```
   public TransformGroup tgtubo;
   public TransformGroup objdeTroca;
   public TransformGroup tgnodulo;

   public Transform3D t3dem;
   public Transform3D t3dembolo;
   public Transform3D t3dag;
   public Transform3D t3dagulha;
   public Transform3D t3dtb;
   public Transform3D t3dtubo;
   public Transform3D objTrans;
   public Transform3D t3dnodulo;

   public Group seringa;
   public Group agulha;
   public Group tubo;
   public Group nodulo;
   private javax.swing.JButton btnPuncao = null;
   private javax.swing.JButton btnTroca = null;
   public int codTreinamento,codAluno;
   boolean flag = true;

   ColisaoNodulo cn;

   public Exame(java.awt.Frame frame, business.Imagem imagem,int cod,int
treina)
       { // TODO AQUI VEM OS OBJ IMAGENS
          codAluno = cod;
          codTreinamento = treina;

          System.out.println("EXAME treinamento "+codTreinamento);
          System.out.println("EXAME aluno "+ codAluno);
          ConnectionManager.ConnectDB("bdavaliacao");
          insereTreina();
          ConnectionManager.CloseDB();
          double size = Double.parseDouble(imagem.getSize());
          //double sizeX= Double.parseDouble(imagem.getSizeX());
          //double sizeY= Double.parseDouble(imagem.getSizeY());
          //double sizeZ= Double.parseDouble(imagem.getSizeZ());
          this.setSize(630, 480);
          java.awt.Dimension dimensao = this.getSize();
          java.awt.Point localizacao = new java.awt.Point();
          localizacao.x = (frame.getBounds().width / 2 -
frame.getInsets().left) - (dimensao.width / 2);
          localizacao.y = (frame.getBounds().height / 2 -
frame.getInsets().top) - (dimensao.height / 2);
          this.setLocation(localizacao);

          setClosable(true);
          setIconifiable(true);

          getContentPane().setLayout(null);

          JLabel labellado = new JLabel();
          labellado.setText(resources.Global.bundle.getString("Direito"));
          labellado.setBounds(1,5,140,15);

          JLabel labelCC = new JLabel();
          labelCC.setText(resources.Global.bundle.getString("ImagemCC"));
          labelCC.setBounds(1,20,140,15);
```

```
            javax.swing.JPanel panelCC = new javax.swing.JPanel();
            JLabel labelImagemCC = new JLabel();
            labelImagemCC.setBounds(1,40,150,365);
            labelImagemCC.setIcon(new ImageIcon(imagem.getIdImagemCC()));

            JLabel labelML = new JLabel();
            labelML.setText(resources.Global.bundle.getString("ImagemML"));
            labelML.setBounds(160,20,140,15);

            JLabel labelImagemML = new JLabel();
            labelImagemML.setBounds(160,40,144,365);
            labelImagemML.setIcon(new ImageIcon(imagem.getIdImagemML()));

            btnPuncao = new javax.swing.JButton();
            btnPuncao.setName("btnPuncao");
            btnPuncao.setText("Iniciar Puncao");
            btnPuncao.setMnemonic('P');
            btnPuncao.setBounds(350, 10, 115, 25);

            btnTroca = new javax.swing.JButton();
            btnTroca.setName("btnTroca");
            btnTroca.setText("Troca Mama");
            btnTroca.setMnemonic('M');
            btnTroca.setBounds(460, 10, 115, 25);

            getContentPane().add(labellado);
            getContentPane().add(labelCC);
            getContentPane().add(labelML);
            getContentPane().add(labelImagemCC);
            getContentPane().add(labelImagemML);
            getContentPane().add(btnPuncao);
            getContentPane().add(btnTroca);

            GraphicsConfiguration config =
    SimpleUniverse.getPreferredConfiguration();
            canvas = new Canvas3D(config);
            canvas.setBounds(310, 40, 300, 365);
            getContentPane().add(canvas);
            setup_universo();

            show();

        }

    public void init()
    {
        this.setup_universo();
    }

    public void setup_universo()
    {
        simpleU = new SimpleUniverse(canvas);
        BranchGroup raizSeringa = this.setup_cena();
        BranchGroup raizMama = this.setup_mama();
        simpleU.getViewingPlatform().setNominalViewingTransform();
        simpleU.addBranchGraph(raizSeringa);
        simpleU.addBranchGraph(raizMama);
     }

    public BranchGroup setup_cena()
        {
```

```
  // Cria a raiz para o gráfico "ramo"
  sceneRoot = new BranchGroup();

  // Crie direção luzes e fundo
  BoundingSphere bounds =
    new BoundingSphere(new Point3d(0.0,0.0,0.0), 100.0);

 sceneTG = new TransformGroup();
   sceneTG.setCapability(TransformGroup.ALLOW_TRANSFORM_WRITE);
   sceneTG.setCapability(TransformGroup.ALLOW_TRANSFORM_READ);
 sceneRoot.addChild(sceneTG);

 MouseRotate myMouseRotate = new MouseRotate();
 myMouseRotate.setTransformGroup(sceneTG);
 myMouseRotate.setSchedulingBounds(new BoundingSphere());
 sceneTG.addChild(myMouseRotate);

 MouseTranslate myMouseTranslate = new MouseTranslate();
 myMouseTranslate.setTransformGroup(sceneTG);
 myMouseTranslate.setSchedulingBounds(new BoundingSphere());
 sceneTG.addChild(myMouseTranslate);

 MouseZoom myMouseZoom = new MouseZoom();
 myMouseZoom.setTransformGroup(sceneTG);
 myMouseZoom.setSchedulingBounds(new BoundingSphere());
 sceneTG.addChild(myMouseZoom);

// Carregando objeto e parâmetro de alteração para o mesmo
objTG = new TransformGroup();
objTrans = new Transform3D();
objTG.getTransform(objTrans);
sceneTG.addChild(objTG);

objTGEM = new TransformGroup();
objTGEM.setCapability(TransformGroup.ALLOW_TRANSFORM_WRITE);
objTGEM.setCapability(TransformGroup.ALLOW_TRANSFORM_READ);
t3dem = new Transform3D();
objTGEM.getTransform(t3dem);
sceneTG.addChild(objTGEM);

TransformGroup objTGAG = new TransformGroup();
objTGAG.setCapability(TransformGroup.ALLOW_TRANSFORM_WRITE);
objTGAG.setCapability(TransformGroup.ALLOW_TRANSFORM_READ);
t3dag = new Transform3D();
objTGAG.getTransform(t3dag);
sceneTG.addChild(objTGAG);

TransformGroup objTGTB = new TransformGroup();
objTGTB.setCapability(TransformGroup.ALLOW_TRANSFORM_WRITE);
objTGTB.setCapability(TransformGroup.ALLOW_TRANSFORM_READ);
t3dtb = new Transform3D();
objTGTB.getTransform(t3dtb);
sceneTG.addChild(objTGTB);

  // Montando a cor de fundo
  Color3f bgColor = new Color3f(0.05f, 0.05f, 0.2f);
  Background bgNode = new Background(bgColor);
  bgNode.setApplicationBounds(bounds);
  sceneRoot.addChild(bgNode);

  // Montando as luzes globais
```

```
  // primeiro, define a cor das luzes
  Color3f light1Color = new Color3f(1.0f, 1.0f, 0.9f);
  Vector3f light1Direction  = new Vector3f(4.0f, -7.0f, -12.0f);
  Color3f light2Color = new Color3f(0.3f, 0.3f, 0.4f);
  Vector3f light2Direction  = new Vector3f(-6.0f, 2.0f, 1.0f);
  Color3f ambientColor = new Color3f(0.1f, 0.1f, 0.1f);

  // Segundo, define a luz do ambiente, e agrega ao "ramo"
  AmbientLight ambientLightNode = new AmbientLight(ambientColor);
  ambientLightNode.setInfluencingBounds(bounds);
  sceneRoot.addChild(ambientLightNode);

  // Por último, defina e agrega as luzes direcionais
  DirectionalLight light1
   = new DirectionalLight(light1Color, light1Direction);
  light1.setInfluencingBounds(bounds);
  sceneRoot.addChild(light1);

  DirectionalLight light2
   = new DirectionalLight(light2Color, light2Direction);
  light2.setInfluencingBounds(bounds);
  sceneRoot.addChild(light2);

  vec = new Vector3f();

  Group nodulo = carregaNodulo();
       tgnodulo = new TransformGroup();
       tgnodulo.setCapability(TransformGroup.ALLOW_TRANSFORM_WRITE);
       t3dnodulo = new Transform3D();
       t3dnodulo.setTranslation(new Vector3f(0.0f, -0.1f, 0.1f));
       t3dnodulo.setScale(0.1);
       tgnodulo.setTransform(t3dnodulo);
       tgnodulo.addChild(nodulo);
       tgnodulo.setCollidable(false);
  sceneRoot.addChild(tgnodulo);

 Group agulha = carregaAgulha();
     tgagulha = new TransformGroup();
     tgagulha.setCapability(TransformGroup.ALLOW_TRANSFORM_WRITE);
     tgagulha.setCapability(TransformGroup.ALLOW_TRANSFORM_READ);
     t3dagulha = new Transform3D();
     t3dagulha.setScale(0.2);
     t3dagulha.setTranslation(new Vector3f(-0.5f, 0.04f, 0.06999f));
     tgagulha.setTransform(t3dagulha);
     tgagulha.addChild(agulha);
objTGAG.addChild(tgagulha);

Group tubo = carregaTubo();
    tgtubo = new TransformGroup();
    tgtubo.setCapability(TransformGroup.ALLOW_TRANSFORM_WRITE);
    tgtubo.setCapability(TransformGroup.ALLOW_TRANSFORM_READ);
     tgtubo.setCapability(TransformGroup.ALLOW_COLLIDABLE_WRITE);
    t3dtb = new Transform3D();
    t3dtb.setScale(0.29);
    t3dtb.setTranslation(new Vector3f(-0.5f, 0.5f, 0.06999f));
    tgtubo.setTransform(t3dtb);
    tgtubo.addChild(tubo);
     tgtubo.setCollidable(false);
objTGTB.addChild(tgtubo);

Group embolo = carregaEmbolo();
```

```
          tgembolo = new TransformGroup();
          tgembolo.setCapability(TransformGroup.ALLOW_TRANSFORM_WRITE);
          tgembolo.setCapability(TransformGroup.ALLOW_TRANSFORM_READ);
          tgembolo.setCapability(TransformGroup.ALLOW_COLLIDABLE_WRITE);
          t3dembolo = new Transform3D();
          t3dembolo.setTranslation(new Vector3f(-0.5f, 0.53f, 0.0f)) ;
          t3dembolo.setScale(0.3);
          tgembolo.setTransform(t3dembolo);
          tgembolo.addChild(embolo);
          tgembolo.setCollidable(false);
     objTGEM.addChild(tgembolo);

    btnPuncao.addMouseListener(new java.awt.event.MouseAdapter()
     {
        public void mousePressed(java.awt.event.MouseEvent e)
        {
           if (flag)
            { doMove( new Vector3f(0.0f,0.15f,0) );
              flag = false;
            }
            else
            { doMove( new Vector3f(0.0f,-0.15f,0) );
             flag = true;
            }
          //}
        }
     });
    // canvas.addMouseMotionListener(this);

    return sceneRoot;
}

public BranchGroup setup_mama()
{
    branch = new BranchGroup();
    branch.setCapability(BranchGroup.ALLOW_DETACH);
    branch.setCapability(BranchGroup.ALLOW_CHILDREN_READ);
    branch.setCapability(BranchGroup.ALLOW_CHILDREN_WRITE);
    branch.setCapability(BranchGroup.ALLOW_CHILDREN_EXTEND);

    BoundingSphere bounds = new BoundingSphere(new Point3d (), 100.0);

    objdeTroca = new TransformGroup();
    objdeTroca.setCapability(TransformGroup.ALLOW_TRANSFORM_WRITE);
    objdeTroca.setCapability(TransformGroup.ALLOW_COLLIDABLE_WRITE);
    objdeTroca.setCapability(BranchGroup.ALLOW_DETACH);
    objdeTroca.setCapability(TransformGroup.ALLOW_CHILDREN_READ);
    objdeTroca.setCapability(TransformGroup.ALLOW_CHILDREN_WRITE);
    objdeTroca.setCapability(TransformGroup.ALLOW_CHILDREN_EXTEND);
    Transform3D t3d = new Transform3D();
    t3d.setScale(0.7);
    //t3d.setRotation(new AxisAngle4d(2.0,0.9,1.0, 0.978));
    t3d.setTranslation(new Vector3f(0.1f, -0.1f, 0.0f));
    objdeTroca.setCollidable(false);
    objdeTroca.setTransform(t3d);
    branch.addChild(objdeTroca);

    Scene mama = null;
    ObjectFile objFileloader = new ObjectFile( ObjectFile.RESIZE );
    mama = null;
```

```
        try {
           mama = objFileloader.load( "object/mama_defor.obj");
        }
        catch (Exception e)
          {
           mama = null;
           System.err.println(e);
          }

        BranchGroup b = mama.getSceneGroup();
        b.setCapability(BranchGroup.ALLOW_DETACH);

        Shape3D shape = null;
        shape = (Shape3D) b.getChild( 0 );

        ColisaoMama cd = new
ColisaoMama(shape,codAluno,codTreinamento,cn,this);
        BoundingSphere boundsCollision = new BoundingSphere();
        cd.setSchedulingBounds(boundsCollision);
        b.addChild(cd);

        objdeTroca.addChild(b);

        btnTroca.addMouseListener(new java.awt.event.MouseAdapter()
               {
               public void mousePressed(java.awt.event.MouseEvent arg0)
               {
                 TrocaMama();
               }
               });

        canvas.addKeyListener(new java.awt.event.KeyAdapter()
        {
           public void keyPressed(java.awt.event.KeyEvent evt)
           {
               int keyCode = evt.getKeyCode();
               if (keyCode == KeyEvent.VK_F1)
               {
                   System.out.println("entrou no teclado trocamama"+keyCode);
                   TrocaMama();
               }
           }
        });

        canvas.addKeyListener(new java.awt.event.KeyAdapter()
        {
            public void keyReleased(java.awt.event.KeyEvent evt)
           {}
        });

        canvas.addKeyListener(new java.awt.event.KeyAdapter()
        {
           public void keyReleased(java.awt.event.KeyEvent evt)
           {}
        });

    return branch;
    }

    public void doMove(Vector3f theMove)
    {
```

```
    objTGEM.getTransform( t3dembolo );
    t3dem.setTranslation(theMove);
    t3dembolo.mul(t3dem);
    objTGEM.setTransform(t3dembolo);
}

public void TrocaMama()
{
    Scene scene = null;
    ObjectFile objFileloader = new ObjectFile(ObjectFile.RESIZE);
    try
    {
       scene = objFileloader.load("object/mama_defor6.obj");
    }
    catch (Exception e)
    {
       scene = null;
       System.err.println(e);
    }
    int num;
      Shape3D shape = null;
      BranchGroup b1 = scene.getSceneGroup();
      shape = (Shape3D) b1.getChild( 0 );
      num=branch.numChildren();
      objdeTroca.removeChild(0);
      objdeTroca.addChild(b1);

}

/* CARREGAR NÓDULO */
private Group carregaNodulo()
{
    Scene scene = null;
    ObjectFile objFileloader = new ObjectFile(ObjectFile.RESIZE);

    try
    {
       scene = objFileloader.load("object/esfera.obj");
    }
    catch (Exception e)
    {
       scene = null;
       System.err.println(e);
    }

    BranchGroup branchGroup = scene.getSceneGroup();
    Shape3D shape = null;
    shape = (Shape3D) branchGroup.getChild( 0 );

    ColisaoNodulo cn = new ColisaoNodulo(shape,codAluno,codTreinamento);
    BoundingSphere bounds = new BoundingSphere();
    cn.setSchedulingBounds(bounds);
    branchGroup.addChild(cn);

return branchGroup;
}

private Group carregaAgulha()
{
    Scene scene = null;
    ObjectFile objFileloader = new ObjectFile(ObjectFile.RESIZE);
```

```
    try
    {
       scene = objFileloader.load("object/agulha.obj");
    }
    catch (Exception e)
    {
       scene = null;
       System.err.println(e);
    }

    BranchGroup branchGroup = scene.getSceneGroup();
    Shape3D shape = null;
    shape = (Shape3D) branchGroup.getChild( 0 );
    shape.setCapability(Node.ALLOW_LOCAL_TO_VWORLD_READ);


return branchGroup;
}

private Group carregaEmbolo()
{
    BranchGroup branchGroup = new BranchGroup();

    Scene scene = null;
       ObjectFile objFileloader = new ObjectFile(ObjectFile.RESIZE);
       try
       {
          scene = objFileloader.load("object/pistao.obj");
       }
       catch (Exception e)
       {
          scene = null;
          System.err.println(e);
       }

       branchGroup.addChild(scene.getSceneGroup());
       return branchGroup;
}

private Group carregaTubo()
{
    BranchGroup branchGroup = new BranchGroup();

    Scene scene = null;
       ObjectFile objFileloader = new ObjectFile(ObjectFile.RESIZE);
       try
       {
          scene = objFileloader.load("object/tubo.obj");
       }
       catch (Exception e)
       {
          scene = null;
          System.err.println(e);
       }

       branchGroup.addChild(scene.getSceneGroup());
       return branchGroup;
}

/* * * *    I N S E R E C O O R D ( )    M É T O D O   * * * * * */
/* Insere as coordenadas de cada movimento na tabela coordenada */
```

```
       public boolean insereTreina() {
           try {
             String insertSQL = "INSERT INTO
treinamento(codAluno,codTreinamento) VALUES " + "(" + codAluno +","+
codTreinamento +")";
            System.out.println("insertSQL");
            System.out.println(insertSQL);
             ConnectionManager.stmt.executeUpdate(insertSQL);
            ConnectionManager.con.commit();
             System.out.println("Treinamento inseridoo!");
             return true;
           }
           catch(SQLException e) {
             try {
               if (e.getErrorCode() == -193) {
                System.out.println("Já existe esta coordenada cadastrado!" +
e.getErrorCode());
                 return false;
               }
               else
                   System.out.println(e.getMessage());
             }
             catch(Exception ex) {
              System.out.println("Problema com tratamento de erros!");
               return true;
             }
           }
           return true;
         }

}
```