LEONARDO TÓRTORA DEVIENNE FERRAZ
RENATO KENICHI SAKATA YAMASHITA

DESENVOLVIMENTO DE JOGO ELETRÔNICO PARA REABILITAÇÃO UTILIZANDO UM SENSOR DE SOM E MOVIMENTO (KINECT)

São Paulo
2012

LEONARDO TÓRTORA DEVIENNE FERRAZ
RENATO KENICHI SAKATA YAMASHITA

DESENVOLVIMENTO DE JOGO ELETRÔNICO PARA REABILITAÇÃO UTILIZANDO UM SENSOR DE SOM E MOVIMENTO (KINECT)

Monografia apresentada à Escola Politécnica da Universidade de São Paulo para a obtenção do título de Engenheiro Mecatrônico.

Área de Concentração:
Engenharia Mecatrônica

Orientador:
Professor Doutor Fabrício Junqueira

São Paulo
2012

FICHA CATALOGRÁFICA

Ferraz, Leonardo Tórtora Devienne
Desenvolvimento de jogo eletrônico para reabilitação utilizando um sensor de som e movimento (KINECT) / L.T.D. Ferraz, R.K.S. Yamashita. -- São Paulo, 2012.
54 p.

Trabalho de Formatura - Escola Politécnica da Universidade de São Paulo. Departamento de Engenharia Mecatrônica e de Sistemas Mecânicos.

1. Realidade virtual 2. Jogos eletrônicos (Desenvolvimento) 3. Sensor 4. Reabilitação I. Yamashita, Renato Kenichi Sakata II. Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia Mecatrônica e de Sistemas Mecânicos III. t.

RESUMO

Neste trabalho será feito o uso da realidade virtual para o desenvolvimento de um jogo eletrônico para reabilitação. Existem muitos pacientes de reabilitação que necessitam de um tratamento muito longo, o que pode ser desestimulante. Os jogos eletrônicos desta categoria têm como objetivo tornar o processo mais eficiente e motivador. A tecnologia avançou ao ponto que se pode interagir com o sistema por meio da captação dos movimentos do corpo humano. Uma das diversas formas de conseguir captar esses movimentos e fazer com que sejam aproveitados dentro de alguma interface disponível ao público é o sensor Kinect. O Kinect é um dispositivo da Microsoft que possui sensores ópticos e de som. Uma de suas vantagens é sua capacidade de comunicação com computadores domésticos, o que permite a criação de *softwares* para uso próprio. Como o sensor Kinect já possui uma interface própria para jogos, pode-se utilizar esta função para o auxílio de pacientes em reabilitação por meio da movimentação de diversas partes do corpo da pessoa, de acordo com a área que se deseja tratar. Um exemplo de jogo proposto é o Pong, que simula um tênis de mesa, em que cada jogador controla uma raquete movimentando uma determinada parte do corpo para cima e para baixo, afim de não permitir que a bola passe.

Palavras-chave: Realidade virtual. Jogos eletrônicos (Desenvolvimento). Sensor. Reabilitação.

ABSTRACT

In this project, virtual reality will be applied in the development of an electronic game to rehabilitation. There are many rehabilitation patients who need a very long treatment, which can be discouraging. This category of electronic games has an objective to make the process more efficient and more encouraging. Technology has advanced to the point that it is possible to interact with the system by means of capturing the movements of the human body. One of the several ways of capturing these movements and making them usable within an interface available to the public is the sensor Kinect. The Kinect sensor is a Microsoft device which has sound and motion sensors. One of its advantages is its capacity of communication with home computers, which allows the creation of softwares for personal use. As the system that will be used already has an interface focused on gaming, this function can be utilized to help patients in rehabilitation. With it it's possible to interact with electronic games by means of voice or the movement of many body parts of the person, according to the area that needs to be treated. An example of a proposed game is Pong, which simulates table tennis, in which each player controls a paddle by moving an assigned body part up and down, so as not to allow the ball to pass.

Keywords: Virtual reality. Electronic games (Development). Sensor. Rehabilitation.

LISTA DE ILUSTRAÇÔES

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

Muitas pessoas precisam passar por algum tipo de reabilitação como consequência de diversas doenças ou de algum acidente. Essa reabilitação consiste de diversas sessões de fisioterapia e isto pode se tornar um fator de desmotivação ao paciente.

Os pacientes em sessões de reabilitação precisam realizar movimentos específicos para que a terapia surta algum efeito. O uso de consoles domésticos junto a esses tratamentos aumenta a motivação do paciente para realizar o tratamento e o índice de acerto dos movimentos feitos por ele (CHANG; CHEN; HUANG, 2011).

Muitos pacientes, principalmente os com dores crônicas, podem, por medo de movimentar o membro a ser reabilitado, evitar usar o membro em questão. O principal objetivo da reabilitação é fazer com que os pacientes realizem atividades físicas para dar-lhes a confiança necessária para que eles voltem às suas atividades normais (HASENBRING, 2000, *apud* SCHÖNAUER et al., 2011).

A reabilitação de pacientes com deficiências motoras necessita que o paciente realize movimentos específicos durante a terapia (GAMA et al., 2012). Por este motivo a escolha de um sistema que possa captar os movimentos dos pacientes é necessária.

Graças à evolução tecnológica é possível usar *videogames* domésticos como ferramentas para reabilitação (LANGE et al., 2011). Acessórios como o Kinect, o Wii e o PlayStation Move podem captar os movimentos de pessoas e permitir que esses movimentos interajam com o jogo.

O dispositivo Wii capta os movimentos dos seus próprios controles e com esses dados estima o movimento da pessoa. Esse método ainda é muito impreciso, pois pode não condizer com o movimento real do usuário. (LANGE et al., 2011).

É neste contexto que aparece o Kinect, que é capaz de captar o movimento de diversas partes do corpo de uma pessoa simultaneamente. Ele pode ajudar os pacientes a realizarem as sessões de reabilitação ao mesmo tempo em que permite que os movimentos dos pacientes sejam os mais naturais possíveis.

O projeto de um *software* efetivo para o Kinect pode, além de aumentar a motivação do paciente e a eficácia da reabilitação, facilitar o trabalho tanto do paciente

quanto do fisioterapeuta, pois o paciente pode aprender de forma mais natural os movimentos corretos da terapia (GAMA et al., 2012).

Apesar de que neste trabalho é usado somente o Kinect para detectar os movimentos do paciente, ele também pode ser usado com outros sistemas de medição, como sensores inerciais, para melhorar a precisão dos dados adquiridos em comparação aos coletados apenas com o Kinect (BÓ; HAYASHIBE; POIGNET, 2011).

1.1 Objetivo

O objetivo deste trabalho é projetar e implementar um jogo utilizando o sensor de som e movimento Kinect para ser utilizado em reabilitação.

Para atingir esse objetivo foram estabelecidas algumas metas:

- Familiarizar-se com a linguagem C#;
- Analisar os códigos do SDK do Kinect e compreendê-los;
- Boa gestão de projeto e integração do código.

1.2 Estrutura do trabalho

Primeiro é feita uma revisão dos conceitos necessários para desenvolvimento do projeto. Em seguida são usados estes conceitos para a elaboração do projeto de um *software* para reabilitação e sua implementação. E por fim é discutida a importância do projeto nas conclusões. No apêndice, é apresentada uma revisão sobre UML com alguns conceitos necessários para compreender a parte de projeto. A estrutura do trabalho é:

- A teoria de reabilitação;
- O funcionamento do Kinect;
- Projeto;
- Implementação;
- Conclusões.

2 REVISÃO BIBLIOGRÁFICA

2.1 Reabilitação

Para criar um jogo interessante para o paciente e que possa ser útil, a *Creative Media & Behavioral Health Center* (CM&BHC) criou um método heurístico que pode ajudar na criação de novos jogos e na avaliação de protótipos. Para tanto, deve-se verificar se o protótipo responde positivamente aos requisitos da Tabela 1 (GOTSIS et al, 2012):

Tabela 1 – Método para projeto e avaliação de protótipos (GOTSIS et al., 2012)

Ideias	Perguntas
Desafio cognitivo	É suficientemente desafiador?
	Consegue se adaptar às habilidades individuais?
	O jogo é envolvente?
Regulagem emocional	É divertido?
	Pode se adaptar a mudanças de humor?
	Fornece um efeito saudável?
Engajamento dialético	O que ele promove?
	É persuasivo?
	Estimula o diálogo interno?
Gratificação somática	As interações físicas parecem naturais?
	O corpo está conectado ao cérebro e à mente?
	As interações físicas são recompensadoras?
Validade socioecológica	Interações sociais são promovidas?
	As regras sociais são respeitadas?
	Onde e como o jogo será usado?
Totalidade semiótica	A experiência é significativa?
	As interações têm algum propósito?
	Como o jogo deve ser entendido?

A CM&BHC é uma unidade de pesquisa formada pela *School of Cinematic Arts* e pela *Keck School of Medicine* e tem como um de seus objetivos melhorar tratamentos por meio da mídia e de tecnologias emergentes (GOTSIS et al., 2012).

Todas as condições são necessárias, porém, para este projeto, algumas têm uma prioridade maior que as outras. São elas:

- Conseguir se adaptar às habilidades individuais
- Ser divertido
- Permitir interações físicas naturais
- Interações terem algum propósito associado

Existem diversas pesquisas feitas sobre o uso do Kinect e grande parte delas apresentaram melhoras nos resultados obtidos quando este acessório foi utilizado, como um aumento no acerto de movimentos corretos durante as sessões de reabilitação (CHANG; CHEN; HUANG, 2011) e uma calibração melhor do equipamento ao captar os movimentos do paciente (BÓ; HAYASHIBE; POIGNET, 2011).

Os *softwares* projetados para ajudar na reabilitação podem, com a ajuda do Kinect, motivar o paciente a continuar o tratamento ou mesmo guiá-lo para que este realize os movimentos corretos durante as sessões.

Estudos feitos em outros trabalhos mostram que os pacientes, ao seguirem os movimentos apresentados durante uma sessão realizada com o Kinect, apresentaram um índice de acerto maior dos que os não utilizaram de nenhuma tecnologia (CHANG; CHEN; HUANG, 2011). A Figura 1 mostra um dos resultados obtidos por um paciente durante as sessões de reabilitação. Nota-se que durante as sessões em que o Kinect é utilizado (*intervention*) o número de movimentos corretos é muito maior do que quando os exercícios são feitos sem o auxílio da tecnologia (*baseline*).

Estes índices de acertos maiores podem estar ligados à motivação do paciente em realizar os exercícios, pois em alguns estudos, diversos pacientes relataram grande satisfação ao experimentarem os jogos (LANGE et al., 2011).

Para distinguir quais movimentos são os corretos e quais são os incorretos, pode ser usado o próprio Kinect. Além disso, também se pode implementar funções que ensinem o paciente sobre como executar os movimentos corretos (GAMA et al., 2012).

O uso do Kinect possibilita não apenas a criação de novos modos de se praticar e monitorar os exercícios como também trouxe a possibilidade de sua integração com outros sistemas de captação de movimentos.

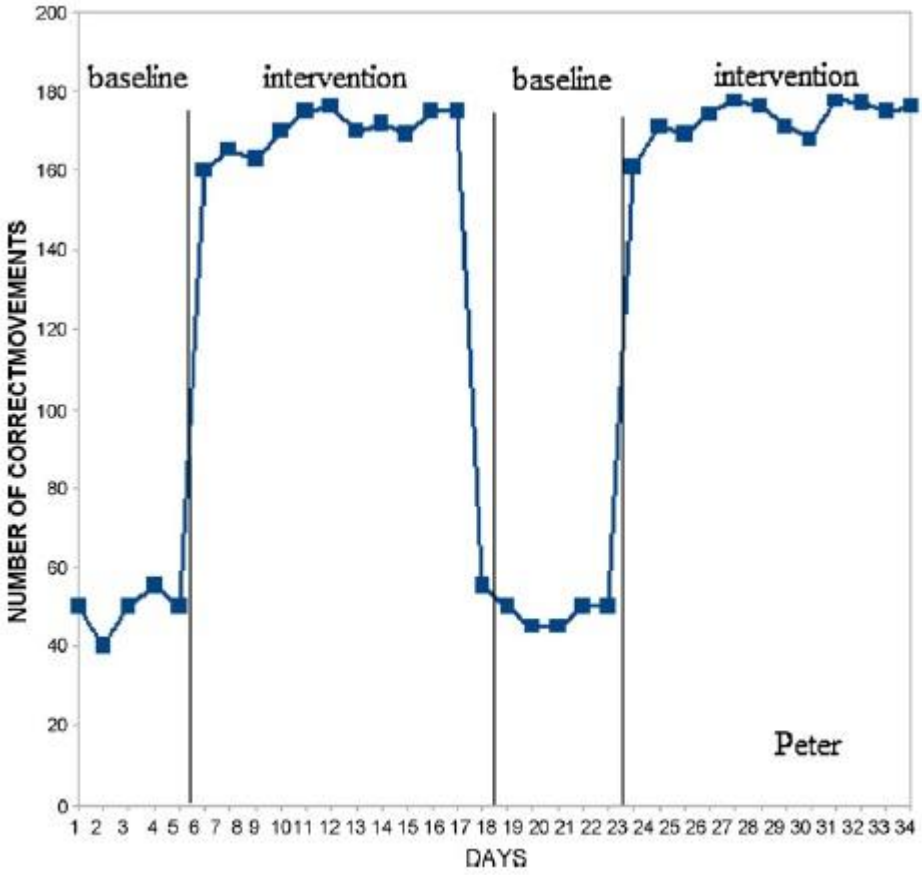


Figura 1 – Número de movimentos corretos realizados pelo paciente por sessão de reabilitação (CHANG; CHEN; HUANG, 2011).

O Kinect já foi usado em conjunto com um sistema de sensores inerciais para estimar a posição das juntas da pessoa. Neste procedimento o Kinect pode ser usado para estimar a posição das juntas e seus ângulos para que depois estes valores sejam usados pelos sensores inerciais. Isso permite uma calibração nos sensores, fornecendo-lhes parâmetros iniciais para a iniciação do algoritmo e a calibração dos sensores em tempo real durante a sessão (BÓ; HAYASHIBE; POIGNET, 2011).

Estes novos métodos estimulam a busca de não só novos usos para o Kinect, mas também a busca de novos sistemas de captura de movimentos para usar na reabilitação de pacientes. Nesse caso o sistema do Kinect pode ser usado como base de comparação para avaliar os resultados obtidos do sistema projetado (SCHÖNAUER et al., 2011).

A complexidade desejada para o jogo depende dos encarregados por ele e do equipamento disponível. Pode-se usar um sistema que capture e utilize os movimentos do corpo como um todo e os use em um jogo relativamente simples cujo objetivo é recolher itens durante o percurso (LANGE et al., 2011) ou até mesmo colocar o paciente em um ambiente virtual mais elaborado onde existe um nível de dificuldade maior em relação aos movimentos do corpo (SCHÖNAUER et al., 2011).

2.2 Kinect

O Kinect (Figura 2) é um aparelho o qual o público tem acesso relativamente fácil e que pode ser usado sem que o paciente possua o *videogame* Xbox 360. Ele pode ser usado em conjunto com qualquer computador doméstico, desde que este possua uma entrada USB e uma versão do Windows 7 ou superior.

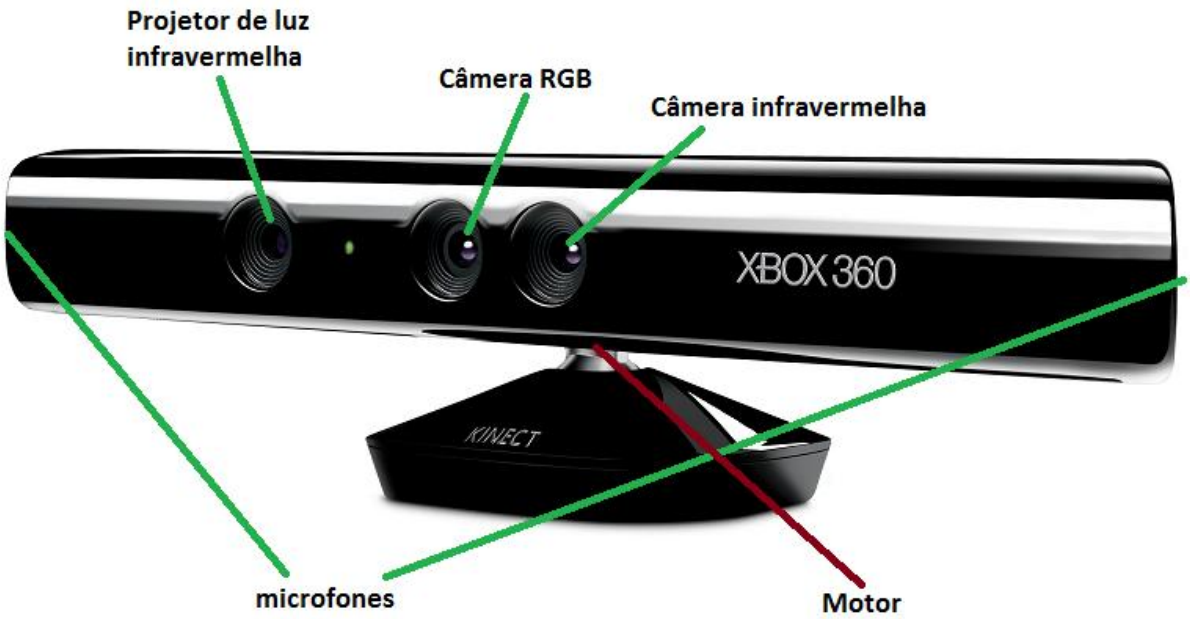


Figura 2 – Kinect e seus componentes (Paula, 2011)

O *software* pode ser executado e desenvolvido em computadores domésticos com o auxílio de um *kit* de desenvolvimento distribuído gratuitamente pela própria

Microsoft, o Kinect SDK, compatível com a plataforma de desenvolvimento Microsoft Visual Studio, que é usada neste projeto.

O Kinect é uma ferramenta que permite captar os movimentos de uma pessoa e utilizá-los em uma interface no computador como, por exemplo, em um jogo. Ele pode identificar um jogador de duas maneiras (LEYVAND et al., 2011):

- *Biometric sign-in*, onde o sistema aprende a aparência do jogador e o reconhece toda vez que ele está jogando;
- *Session*, onde o sistema lembra quem é quem durante uma sessão de jogo.

Para relacionar um jogador a um determinado perfil, o sistema trabalha com três técnicas: reconhecimento facial, reconhecimento da cor da roupa e estimativa de altura (LEYVAND et al., 2011).

A captura do formato do esqueleto é feita com nós, que representam partes do corpo do jogador. O corpo virtual então é criado tendo como base estes nós (Figura 3).

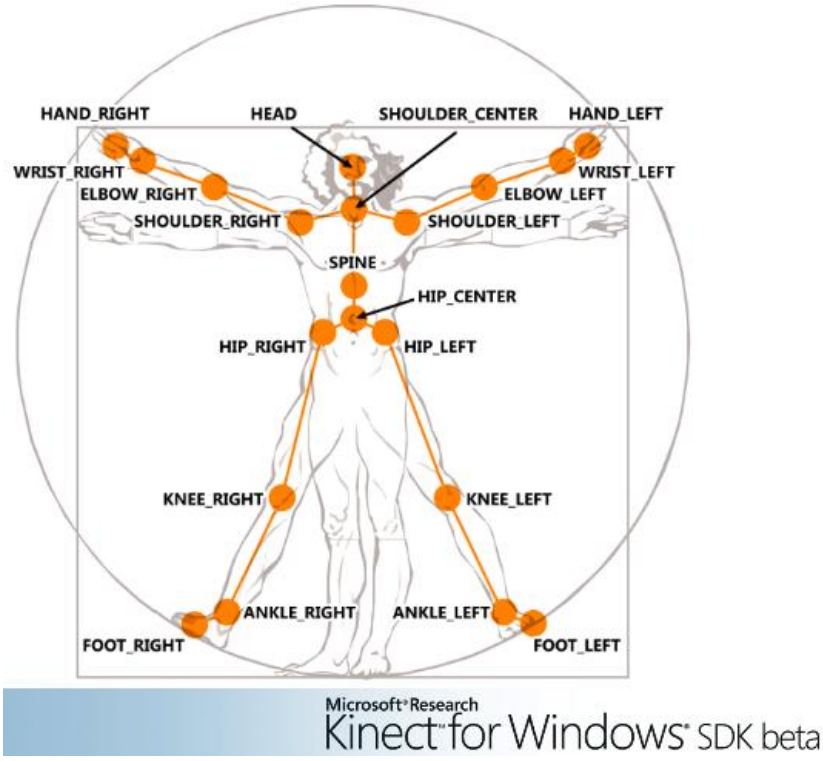


Figura 3 – Nós criados para a reconstituição do corpo (MICROSOFT, 2011)

A Figura 4 apresenta um exemplo de como o esqueleto é captado.

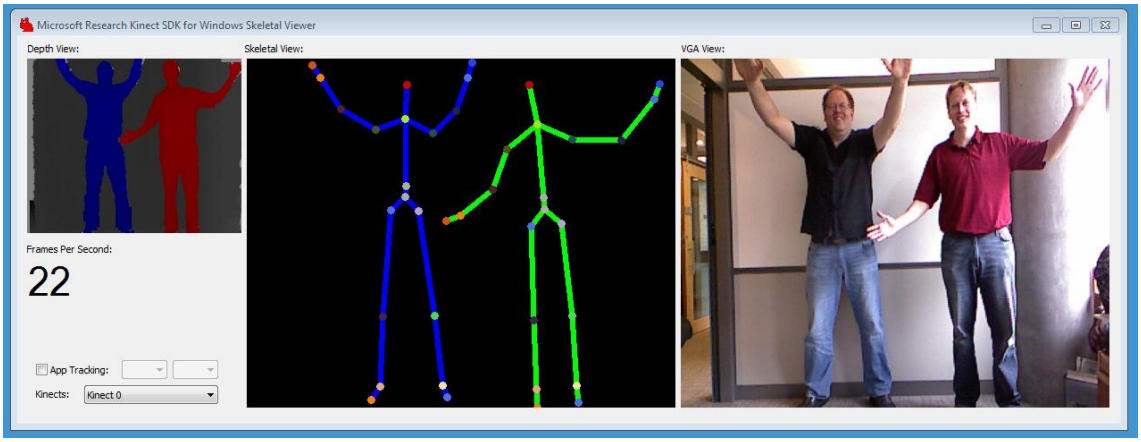


Figura 4 – Captura dos nós (centro) e uso do sensor de profundidade para diferenciação de corpos (esquerda) (MICROSOFT, 2012)

Quando a reconstituição do esqueleto não pode ser feita de forma perfeita, ou porque o corpo do jogador não é captado por inteiro, ou porque o jogador está numa posição que não permite a distinção entre nós etc., o sistema do Kinect extrapola o formato do esqueleto, ou seja, ele assume certas posições para os nós do corpo do jogador.

O *hardware* do Kinect é formado por um projetor de luz infravermelha, uma câmera infravermelha, uma câmera RGB, um motor e um conjunto de microfones (Figura 2). Estes dispositivos conseguem processar e devolver os seguintes dados para o computador (PAULA, 2011):

- *Image Stream*: é a imagem da câmera RGB ou da câmera infravermelha;
- *Depth Stream*: imagens capturadas pela câmera infravermelha. O padrão infravermelho é projetado (e capturado pela câmera infravermelha) e a deformação neste padrão é medida. Assim, é possível determinar a profundidade dos objetos;
- *Audio Stream*: com um conjunto de 4 microfones, o Kinect permite a gravação de áudio e reconhecimento da fala.

Cada imagem obtida pelas câmeras RGB e infravermelha é formada por diversos pixels e a cada pixel (da imagem infravermelha) é associado um valor que possibilita identificar a qual pessoa pertence o corpo. Assim, é possível diferenciar entre até seis pessoas, porém apenas pode-se obter a informação do esqueleto de duas pessoas simultaneamente (PAULA, 2011).

O Kinect consegue captar normalmente movimentos em uma distância de 3 a 4 metros. Valores acima destas distâncias são considerados muito distantes para funcionamento normal do sistema. Para valores além de 8 metros os efeitos são desconhecidos (MICROSOFT, 2012).

3 PROJETO

Para tornar a experiência de reabilitação mais confortável, motivadora e mesmo eficiente, foi proposto um jogo simples e de fácil compreensão.

Para alcançar o grau de simplicidade desejado, foi escolhido o jogo no estilo Pong (Figura 5), um dos primeiros jogos de *videogame* da história[1].

Figura 5 – Protótipo de tela para o jogo Pong

A ideia é fazer com que o paciente use até duas juntas do corpo, reconhecidas pelo Kinect e escolhidas por um auxiliar para controlar ambas as raquetes, representadas por duas barras verticais. O objetivo do jogo é não permitir que a bola chegue às laterais da tela.

[1] <en.wikipedia.org/wiki/Pong> Pong – Wikipedia, the free encyclopedia

Outros parâmetros também podem ser ajustados, como é o caso da velocidade da bola, do intervalo de tempo máximo para a sessão e outros que podem ser vistos no detalhamento do projeto.

Alguns requisitos do projeto são:

- Desenvolver um jogo que seja motivador para o processo de reabilitação e que possa trazer melhores resultados que a reabilitação convencional.
- O jogo deve ser desenvolvido com uma interface simples de forma que o paciente e seu auxiliar possam jogar em casa facilmente sem a supervisão do médico ou do fisioterapeuta.
- O jogo deve possuir níveis de dificuldade adequados a uma sessão de reabilitação.

Para o projeto do programa, foram realizados diagramas UML utilizando o *software* StarUML[2]. A seguir serão apresentados todos os diagramas.

3.1 Diagrama de casos de uso

Primeiramente foi realizado o diagrama de casos de uso (Figura 6), que mostra o comportamento do sistema e as ações que podem ser realizadas. Os atores são o “Paciente” (que será a pessoa que jogará o jogo em si) que ficará na posição de jogo (aproximadamente a 3 metros de distância do Kinect) e o “Auxiliar” (que pode ser o médico, ou o fisioterapeuta ou qualquer pessoa que esteja auxiliando o paciente no processo de reabilitação), que ficará no computador e irá incluir o nome do paciente e ajustar os parâmetros do jogo como a velocidade da bola, o tempo de jogo, o tamanho da raquete e a amplitude do movimento da junta do paciente, além de definir quais juntas irão controlar as raquetes e também obter as estatísticas das partidas jogadas pelo paciente e salvá-las. Alguns casos de uso poderão ser realizados tanto pelo “Paciente” quanto pelo “Auxiliar”, por isso eles estão associados ao “Ator”, que representa um usuário genérico do *software*. Nota-se que alguns casos de uso incluem os casos de uso “Detecta movimento” e “Detecta voz”, esses são os casos de uso responsáveis por movimentar a raquete na tela e por reconhecer os comandos de voz do usuário, respectivamente. Observa-se que o “Mover raquete” não inclui o “Detecta

[2] <staruml.sourceforge.net> StarUML the Open Source UML/MDA Platform

voz”, pois mover a raquete com a voz não é a premissa do projeto. Exceto pelo “Mover raquete”, os casos de uso do “Paciente” não incluem o “Detecta movimento”, pois durante o jogo, o paciente estará controlando a raquete com o movimento, sendo muito mais prático executá-los com um comando de voz. Como existe a possibilidade do “Paciente” estar impossibilitado de executar os comandos de voz, eles poderão ser executados pelo “Auxiliar” também.

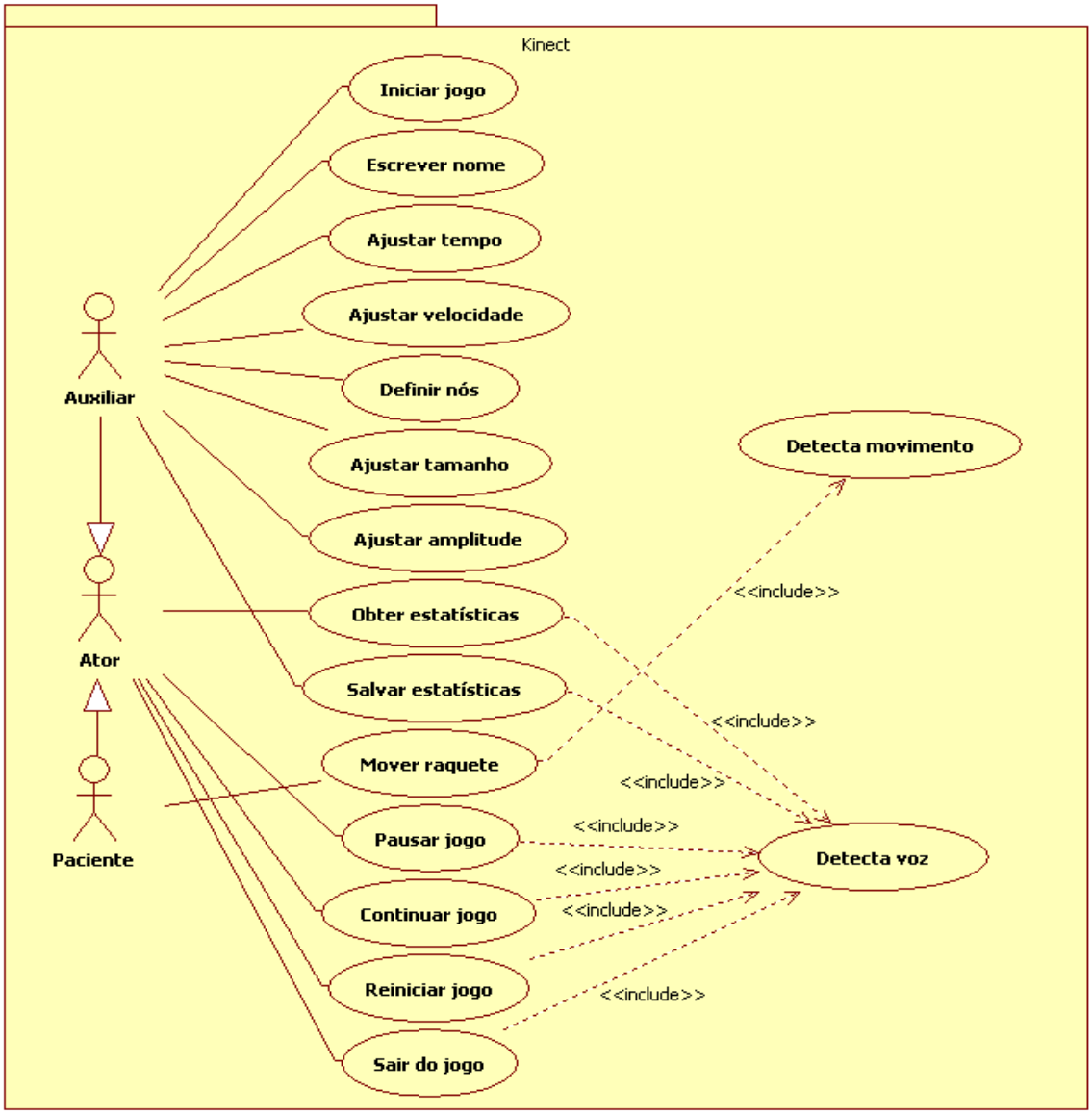
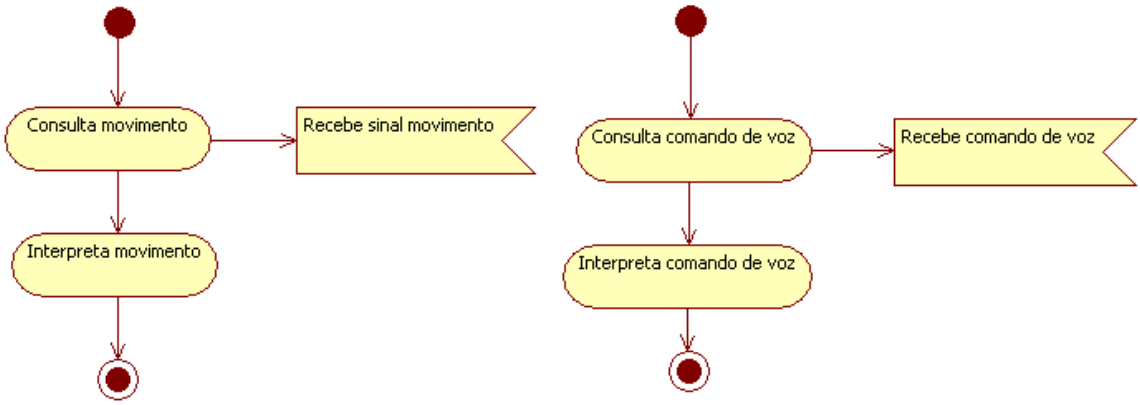


Figura 6 – Diagrama de casos de uso

3.2 Diagramas de atividade

Em seguida foi realizado um diagrama de atividade para cada caso de uso. Esses diagramas mostram seus fluxos de atividades.

Nos dois diagramas da Figura 7, primeiramente o Kinect irá consultar se recebeu um sinal que será um movimento ou um comando de voz. Ao recebê-lo, irá interpretar o comando e realizará a ação correspondente. Se o sinal for um movimento, o Kinect irá mover a raquete adequadamente. Se for um comando de voz, o programa irá buscá-lo na biblioteca e realizará a ação correspondente.



(a) (b)

Figura 7 – Diagramas de atividade: (a) “Detecta movimento”; (b) “Detecta voz”

Ao clicar no botão “Iniciar”, serão carregados todos os parâmetros que foram configurados pelo auxiliar e em seguida a tela do jogo será carregada (Figura 8).

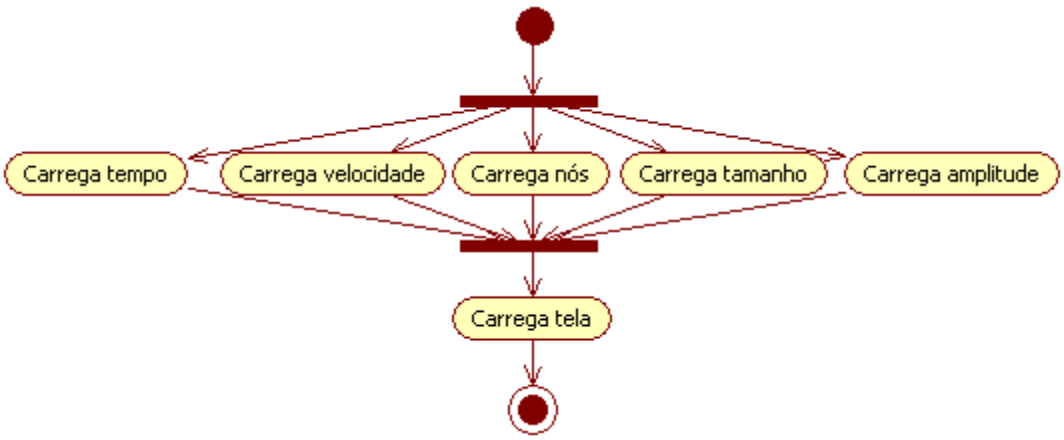


Figura 8 – Diagrama de atividade de “Iniciar jogo”

O diagrama da Figura 9 indica que, estando na tela de jogo, após o Kinect receber um sinal de movimento e interpretá-lo, o programa irá verificar a posição atual da raquete e irá atualizá-la.

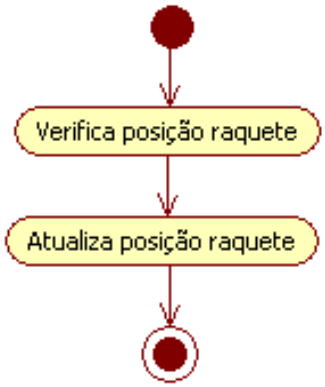


Figura 9 – Diagrama de atividade de “Mover raquete”

Os seis diagramas da Figura 10 representam os casos de uso “Pausar jogo”, “Continuar jogo”, “Sair do jogo”, “Obter estatísticas”, “Salvar estatísticas” e “Escrever nome”, respectivamente, enquanto o da Figura 11-a representa o de “Reiniciar jogo”. Ao pausar o jogo (com um comando de voz) será mostrada uma tela com a opção de continuar o jogo de onde foi pausado (“Continuar”), encerrar o jogo atual e iniciar um novo jogo (“Novo jogo”), mostrar as estatísticas do jogo atual (“Obter estatísticas”) ou sair do jogo (“Sair”). O menu inicial terá 2 botões além dos botões de ajuste dos parâmetros: o “Iniciar” que se pelo menos uma das raquetes estiver habilitada e sua junta correspondente escolhida, o jogo irá começar; e o botão “Sair”, que irá fechar o jogo, além de ter um campo para ser preenchido com o nome do jogador. A tela de estatísticas mostrará as estatísticas da partida e aparecerá ao final de cada uma, ou ao clicar no botão “Obter estatísticas” na tela de pausa, caso se deseje encerrar a partida antes do tempo estipulado previamente. Esta tela possui: a pontuação final, o tempo de duração da partida, e para cada uma das raquetes, os acertos totais, os acertos no centro da raquete e as bolas perdidas. Além dos botões “Novo Jogo” e “Sair”, esta tela também possui o botão “Salvar estatísticas”, que irá salvá-las em um arquivo de texto. Neste arquivo, também serão incluídos o nome do jogador, a data e o horário em que o jogo foi jogado, a velocidade da bola, as juntas escolhidas, os tamanhos das raquetes e as amplitudes do movimento das juntas. Com as estatísticas pode-se observar o quanto

o paciente foi preciso naquela partida e observar sua evolução em relação a partidas anteriores.

(a) (b) (c) (d) (e) (f)

Figura 10 – Diagramas de atividade: (a) “Pausar jogo”; (b) “Continuar jogo”; (c) “Sair do jogo”; (d) “Obter estatísticas”; (e) “Salvar estatísticas”; (f) “Escrever nome”

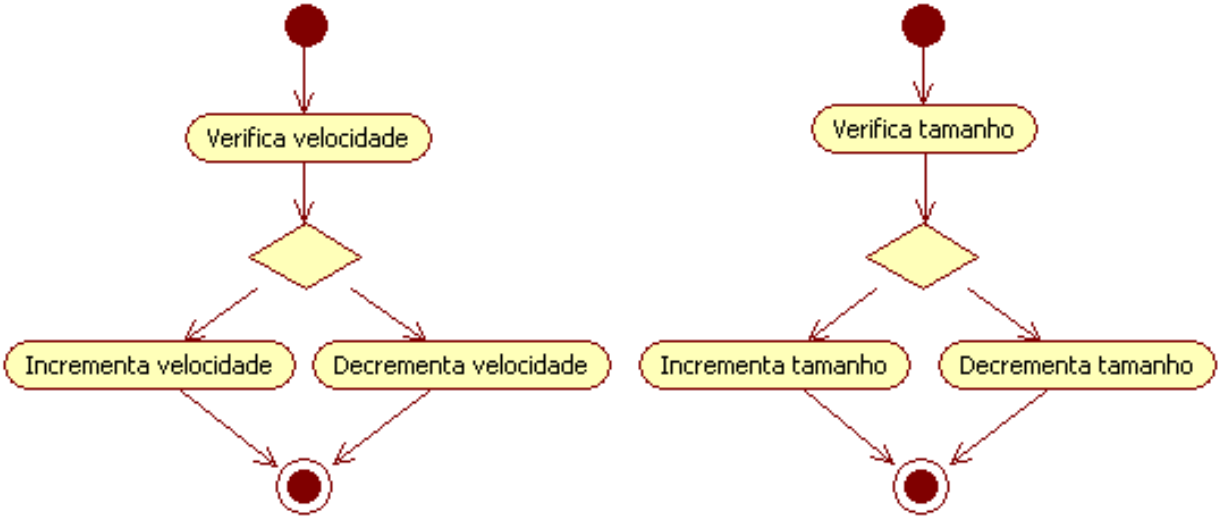
(a) (b)

Figura 11 – Diagrama de atividade: (a) “Reiniciar jogo”; (b) “Definir nós”

Na tela inicial, o auxiliar irá definir os nós das juntas que serão utilizadas pelo paciente na partida (Figura 11-b). Para cada raquete haverá uma *checkbox* que indicará se aquela raquete será utilizada associada a uma junta, caso contrário, a raquete irá ocupar o tamanho da tela e funcionará como uma parede. Além disso, haverá uma *combobox* que servirá para escolher a junta que será associada com aquela raquete. As juntas que poderão ser escolhidas são as das mãos, dos joelhos, ou a dos pés.

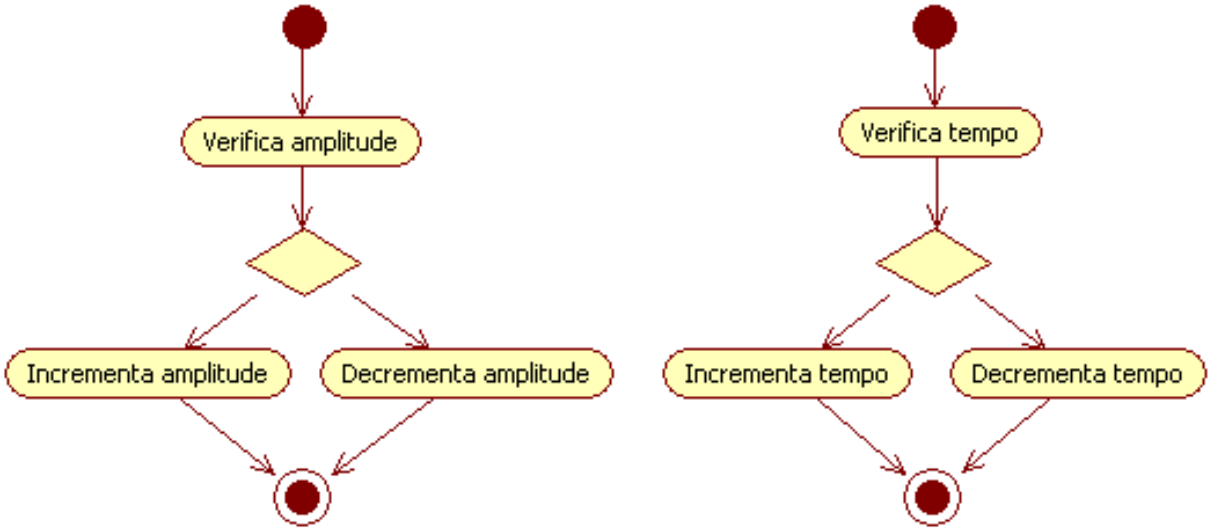
Os diagramas a seguir (Figura 12 e Figura 13) mostram que, na tela inicial, quando o auxiliar for ajustar a velocidade da bola (Figura 12-a), o tamanho da raquete (Figura 12-b), a amplitude do movimento da junta do paciente (Figura 13-a) e o tempo de jogo (Figura 13-b), ele poderá incrementar ou decrementar os valores

correspondentes a cada parâmetro. Primeiro o programa verifica o valor atual e, se for possível, realiza a ação.



(a) (b)

Figura 12 – Diagramas de atividade: (a) “Ajustar velocidade”; (b) “Ajustar tamanho”



(a) (b)

Figura 13 – Diagrama de atividade: (a) “Ajustar amplitude”; (b) “Ajustar tempo”

A velocidade da bola será um valor que irá de 1 a 5, discretizado de 0,5 em 0,5, sendo 5 a velocidade mais rápida; o tamanho da raquete também será um valor que irá de 1 a 5, variando de 0,5 em 0,5, sendo 5 o tamanho maior; e o tempo de jogo irá variar

de 0 a 99 minutos e 50 segundos, variando de 10 em 10 segundos. Quanto à amplitude do movimento, ela vai de 10 a 200 cm variando de 10 em 10 cm, ela define a amplitude máxima que o paciente irá movimentar os braços ou as pernas, quanto maior a amplitude, menos sensível é o movimento da raquete.

3.3 Diagrama de classe

O diagrama de classe (Figura 14) é útil para se ter uma visão estática do projeto. A classe “Jogo” é a classe principal, que contém as classes “DepthImagePoint”, “SpeechRecognitionEngine”, “Bola”, “Rdireita”, “Resquerda”, “JogoConfig”, “Forms” e “Colisao”.

A classe “Colisao” irá verificar se houve uma colisão utilizando a função “checkColisão” e, caso ela ocorra, irá executar a função “processaColisão”, que irá verificar o tipo da colisão e executar a ação correspondente. Uma colisão pode ser de 3 tipos: entre a bola e uma raquete, entre a bola e a borda superior ou inferior e entre a bola e uma borda lateral. Considerando o eixo X na horizontal e o eixo Y na vertical, no primeiro tipo, assim que a bola colidir com a raquete, serão verificadas as componentes X e Y do vetor velocidade da bola (variáveis “velx” e “vely” da classe “Bola”), será feita uma conta com essas velocidades e serão determinadas as novas componentes do vetor velocidade da bola. No segundo caso, quando a bola colidir com a borda superior ou inferior, a componente Y do vetor velocidade da bola terá seu sentido invertido. No terceiro e último caso, quando a bola colidir com uma borda lateral, significa que ocorreu um gol, desse modo, a bola irá sumir e reaparecer no centro da tela, assim como no início do jogo. O valor inicial das componentes X e Y do vetor velocidade da bola é randômico, porém a bola nunca começará numa direção paralela às bordas superior e inferior, pois neste caso ela não mudaria de direção ao colidir com as raquetes e o paciente não necessitaria se mover, tampouco começará numa direção paralela às bordas laterais, pois neste caso ela permaneceria na direção central do campo colidindo somente com as paredes, e nunca com as raquetes.

No placar será mostrado o tempo restante e a pontuação atual do paciente. Se o paciente fizer com que a bola acerte o centro da raquete, a pontuação aumentará 30

pontos, se a bola acertar qualquer outra parte da raquete, ela aumentará 10 pontos e se ele deixar a bola passar, ela diminuirá 5 pontos.

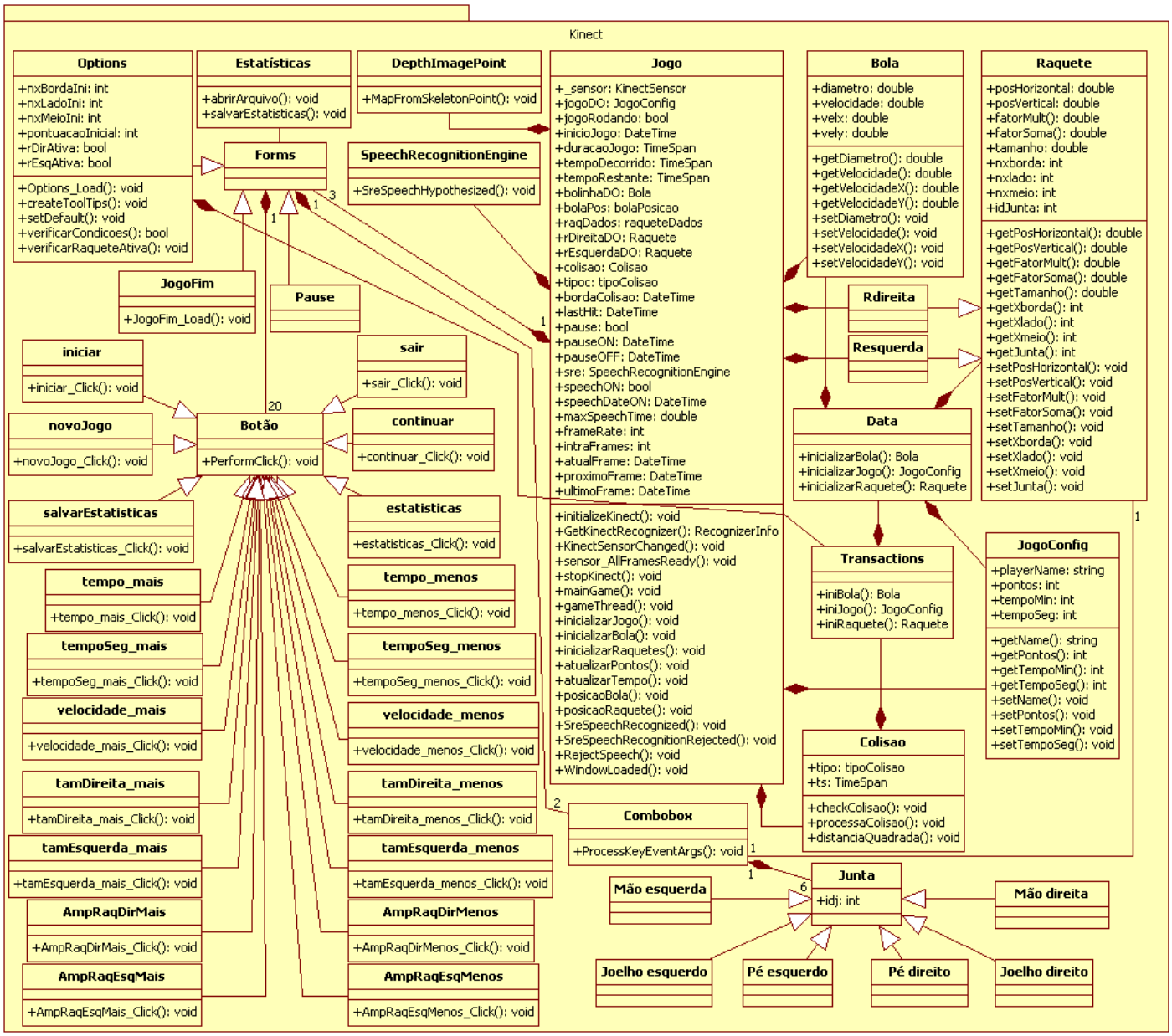


Figura 14 – Diagrama de classe

Além da tela de jogo, o jogo possuirá 3 “Forms”, a da tela inicial (“Options”), a tela de pausa (“Pause”) e a tela de estatísticas que é mostrada no final do jogo (“JogoFim”). Nestes “Forms” existirão 20 botões ao todo. Cada botão possui uma função *click* que o ativa. Também existirão 2 *combobox*, cada uma associada a uma

das raquetes. Em cada uma, poderá ser escolhida uma dentre as 6 juntas para ser associada à raquete.

A classe “Data” cria os objetos “Raquete”, “Bola” e “JogoConfig” e atribui valores para cada um dos seus parâmetros, que serão usados durante a sessão. A classe “Transactions” tem a função de chamar essas funções que criam os objetos para o programa principal.

As variáveis “playerName” e “pontos” da classe “JogoConfig”, a variável “tempoDecorrido” da classe “Jogo” e as variáveis “nxborda”, “nxlado”, “nxmeio” e “idJunta” da classe “Raquete” serão utilizadas para gerar as estatísticas da partida. As outras funções e variáveis relevantes serão explicadas junto com os diagramas de sequência.

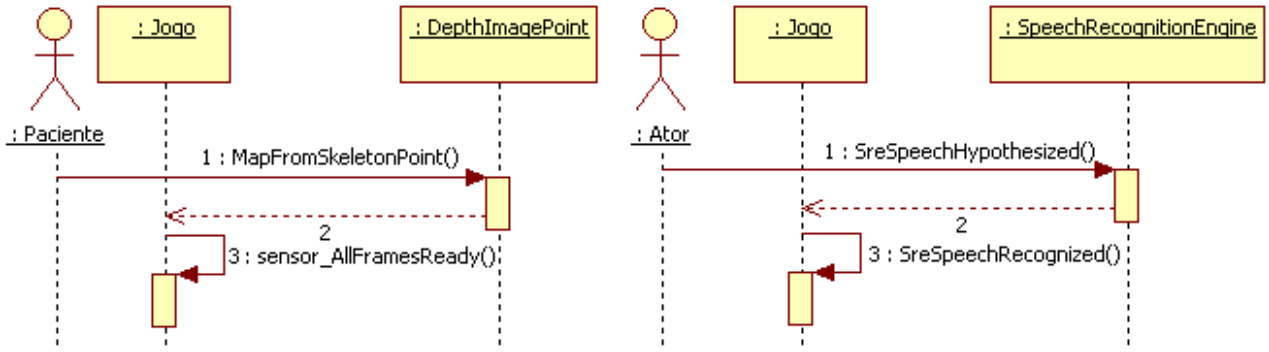
3.4 Diagramas de sequência

Por fim, foram realizados os diagramas de sequência. Esses diagramas são úteis para se ter uma visão dinâmica do projeto.

Os diagramas da Figura 15 representam o que acontece quando o Kinect recebe um sinal. Primeiramente, o “Paciente” ou o “Auxiliar” irá executar um movimento ou um comando de voz. Assim que o Kinect recebe o sinal, através das funções “MapFromSkeletonPoint” ou “SreSpeechHypthesized” ele envia as coordenadas do movimento ou o comando de voz para serem interpretados pelas funções “sensor_AllFramesReady” ou pela “SreSpeechRecognized”, respectivamente. A função “sensor_AllFramesReady” irá chamar o caso de uso “Mover raquete” que será responsável pela movimentação da raquete. A função “SreSpeechHypothesized” irá analisar o comando de voz e buscar na biblioteca o comando mais próximo, ela também atribuirá um valor indicando o quão próximo o comando está da palavra encontrada. Se o valor for alto o suficiente, será chamada a função “SreSpeechRecognized” que executará a ação correspondente àquele comando. Como o caso de uso “Pausar jogo” é executado somente com um comando de voz, ele está representado junto com o “Detecta voz”.

Como vários casos de uso são somente um *click* de um botão, foi realizado somente um diagrama de sequência que representa todos eles (Figura 16). Os casos

de uso representados são o de “Iniciar jogo”, “Continuar jogo”, “Reiniciar Jogo”, “Sair do jogo”, “Obter estatísticas” e “Salvar estatísticas”. As ações desses botões já foram detalhadas nos diagramas de atividade.



(a) (b)

Figura 15 – Diagramas de sequência: (a) “Detecta movimento”; (b) “Detecta voz”

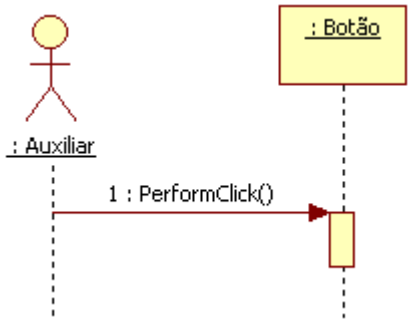


Figura 16 – Diagrama de sequência de um botão

Quando o jogador estiver na tela de jogo e executar um movimento, a função “sensor_AllFramesReady” da classe “Jogo” irá chamar a função “getPosVertical” da classe “Raquete” e retornar a posição para a classe “Jogo”. A função “posicaoRaquete” irá calcular a nova posição da raquete e por fim, a função “setPosVertical” irá atualizar a posição da raquete na tela. Este diagrama é representado na Figura 17.

O diagrama representado na Figura 18 é o “Definir nós”. Nele, ao clicar na *combobox* irão aparecer as 6 opções de junta na tela. Ao clicar em uma junta, o ID correspondente a ela será registrado na variável “idJunta” da classe “Rdireita” ou da classe “Resquerda” dependendo de qual *combobox* foi clicada.

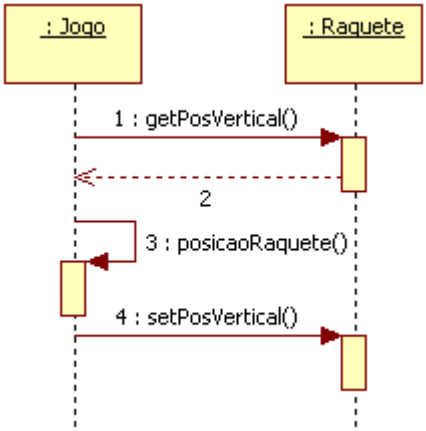


Figura 17 – Diagrama de sequência de “Mover raquete”

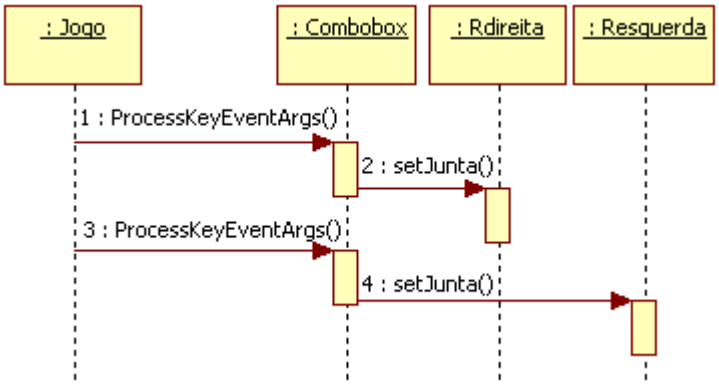
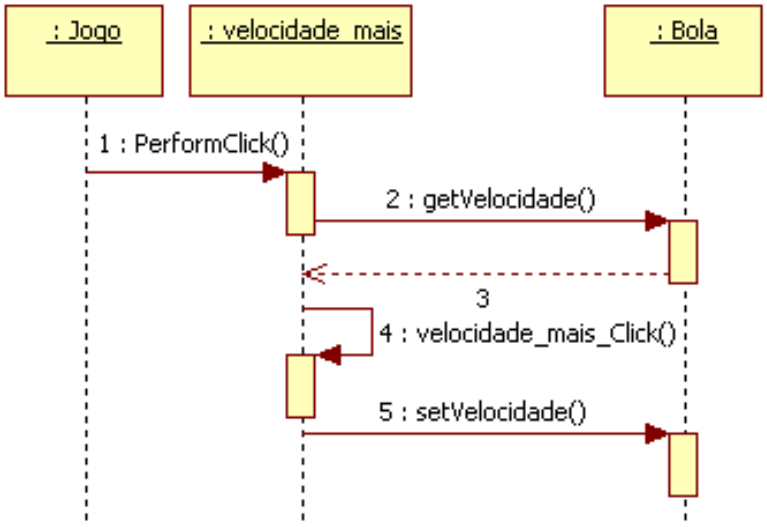


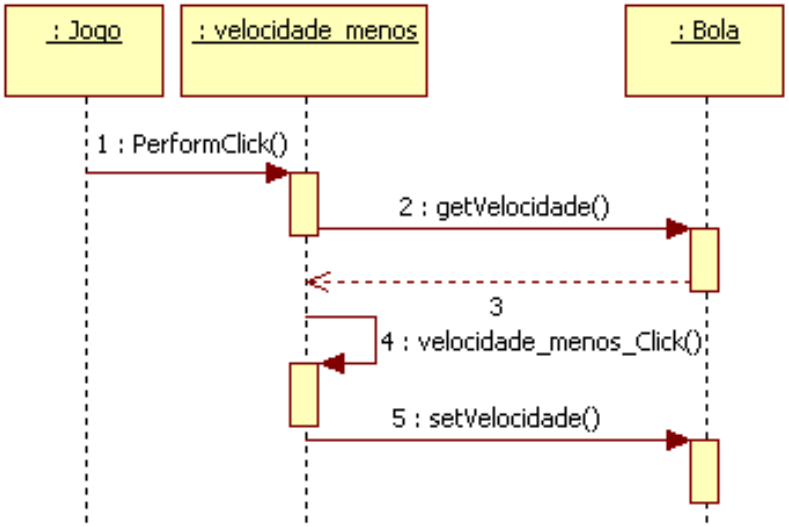
Figura 18 – Diagrama de sequência de “Definir nós”

Na tela inicial de opções terá um botão “+” (“Aumentar”) e um botão “-” (“Diminuir”) dos lados do valor de cada parâmetro. Esses parâmetros são: velocidade da bola (Figura 19), tamanho da raquete (Figura 20), amplitude do movimento (Figura 21) e tempo de jogo (Figura 22). Estes diagramas mostram que, ao clicar em um desses botões, a função “PerformClick” chamará o método “get” correspondente ao parâmetro que se deseja alterar. Tendo o valor atual do parâmetro, a função *click* do botão vai verificar se é possível aumentar ou diminuir o parâmetro e caso seja, irá chamar o método “set” para atualizar seu valor. No caso do tempo, podem ser alterados os minutos e os segundos separadamente. No caso dos parâmetros das raquetes, existem funções para a raquete da direita e para a raquete da esquerda. Além disso, no caso da amplitude do movimento, são alterados dois fatores, o “fatorSoma” e o “fatorMult” que são utilizados na conta para alterar a sensibilidade da raquete. Estes fatores transformam o valor da amplitude do movimento da junta do paciente em

centímetros que o auxiliar escolhe para o valor real das coordenadas na tela. É feita uma transformação de coordenadas diferente para cada junta, pois o movimento da raquete deve se adequar ao movimento da junta escolhida.

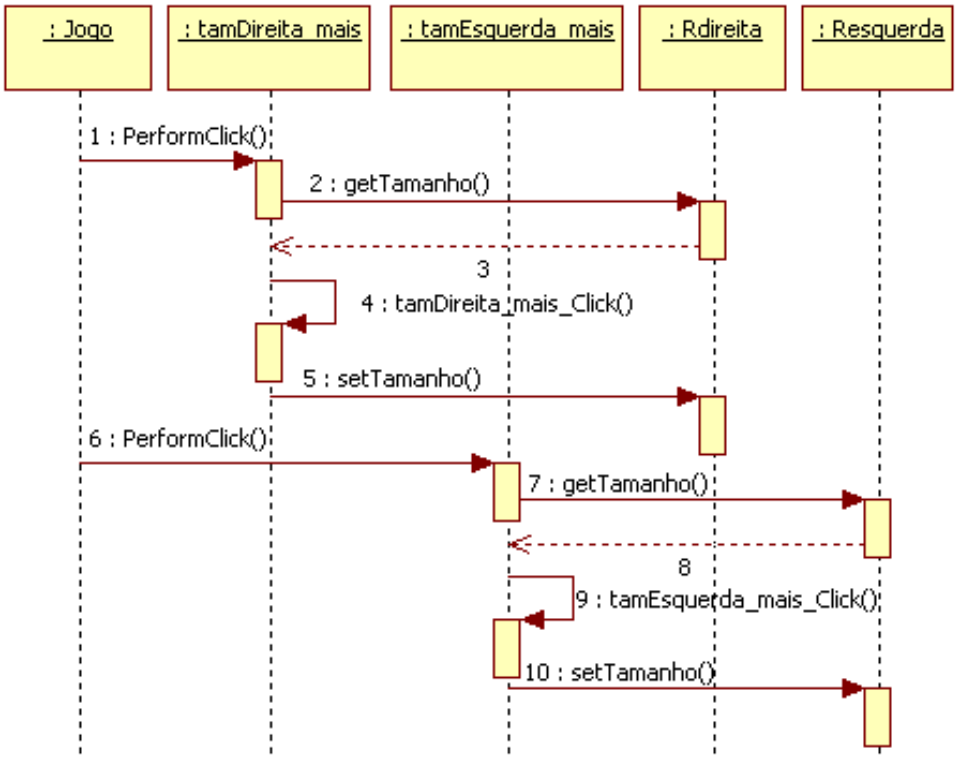


(a)

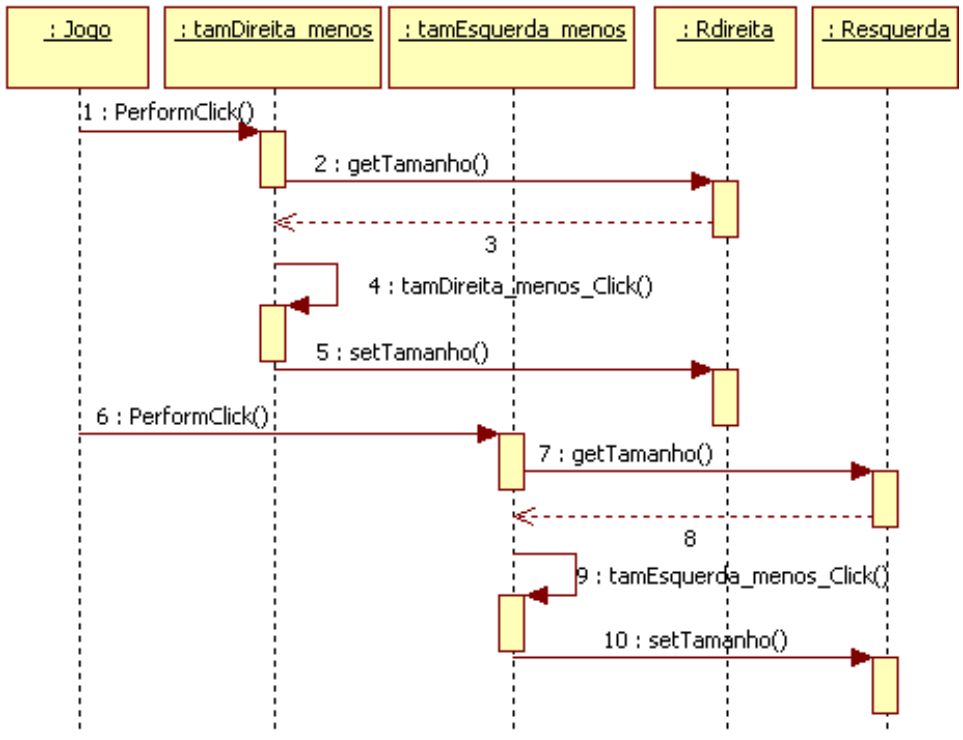


(b)

Figura 19 – Diagramas de sequência: (a) “Aumentar velocidade”; (b) “Diminuir velocidade” da bola

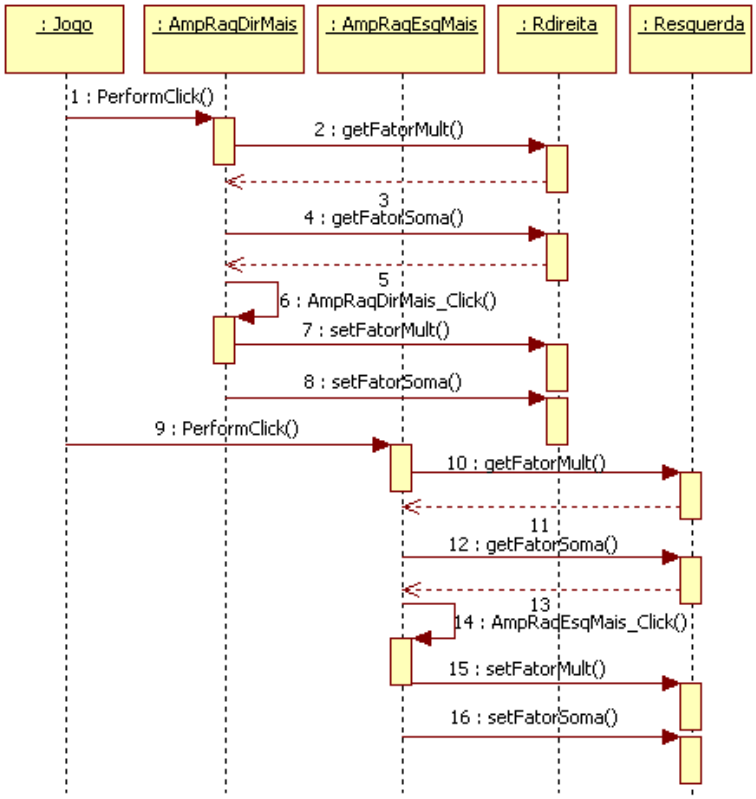


(a)

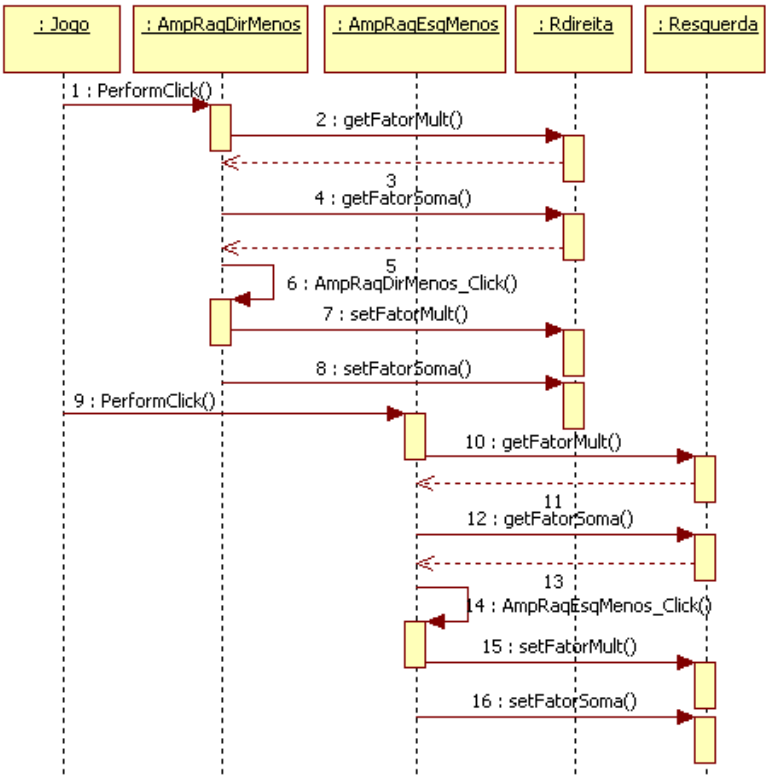


(b)

Figura 20 – Diagramas de sequência: (a) “Aumentar tamanho”; (b) “Diminuir tamanho” da raquete

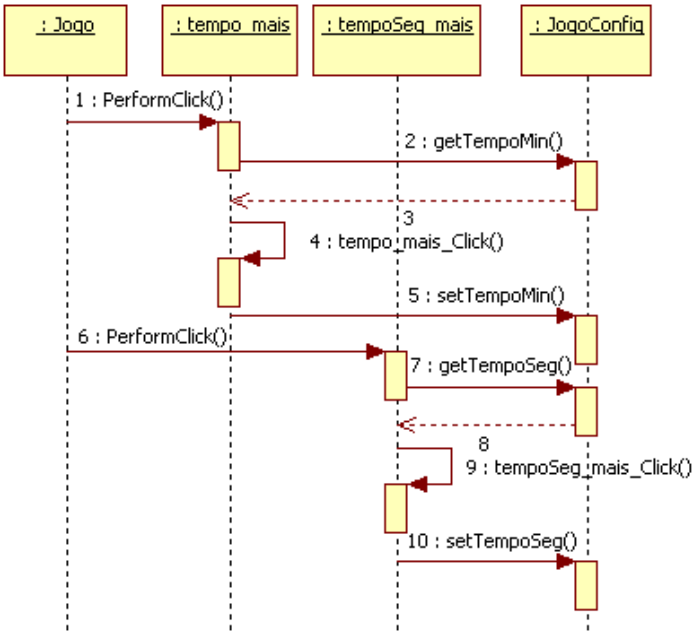


(a)

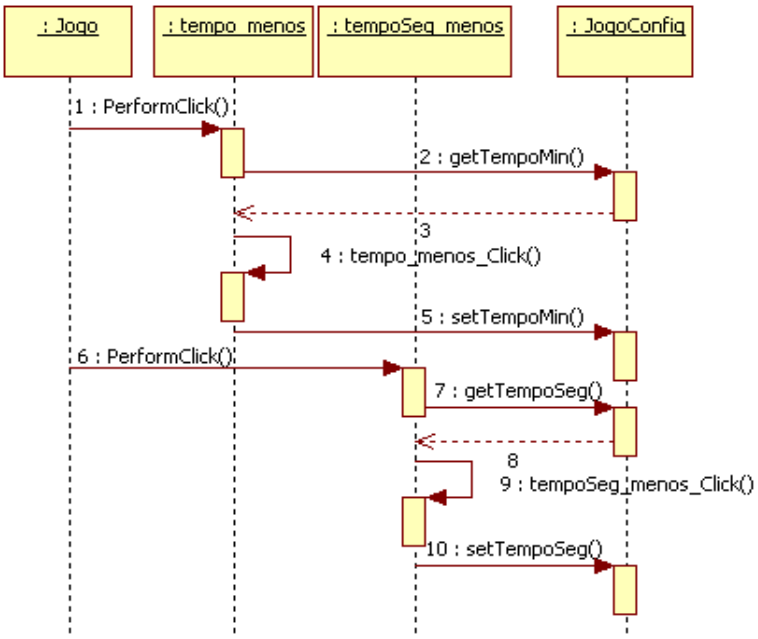


(b)

Figura 21 – Diagramas de sequência: (a) “Aumentar amplitude”; (b) “Diminuir amplitude” do movimento



(a)



(b)

Figura 22 – Diagramas de sequência: (a) “Aumentar tempo”; (b) “Diminuir tempo” do jogo

4 IMPLEMENTAÇÃO

Para implementar o projeto foi utilizado o programa Microsoft Visual Studio 2010 juntamente com o Kinect for Windows SDK v1.5.

A Figura 23 mostra a tela principal do jogo Pong (no caso das 2 raquetes estarem habilitadas e cada uma associada a uma junta), contendo a pontuação, o tempo restante de jogo, um ícone de voz que indica se o comando de voz inicial foi aceito e que os outros comandos de voz foram habilitados, as raquetes e a bola.

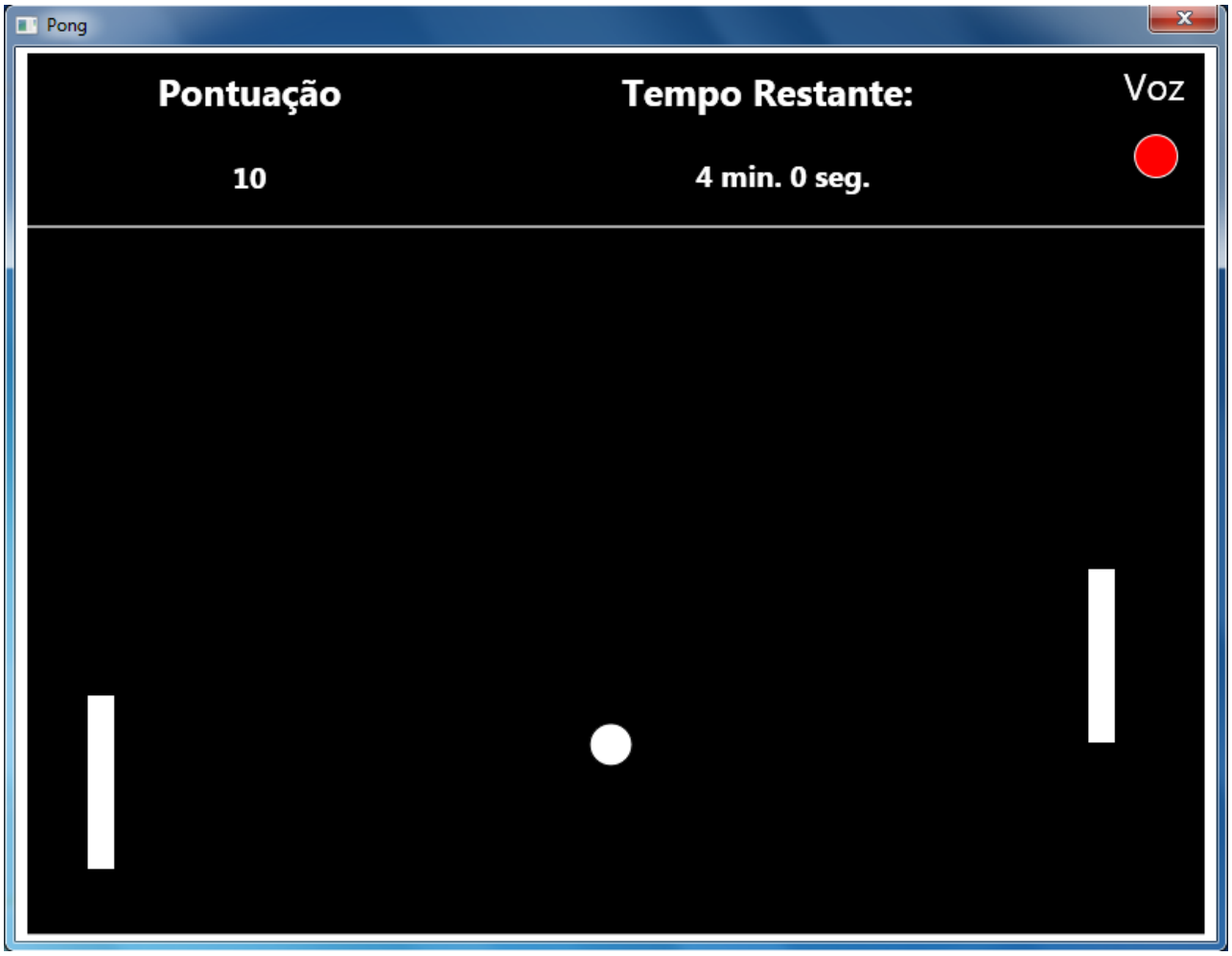


Figura 23 – Tela do jogo Pong com as 2 raquetes habilitadas

Ao abrir o arquivo do Pong, a tela de configuração para o jogo (Figura 24) será mostrada ao usuário.

Nesta tela pode-se configurar os seguintes dados sobre o jogo:

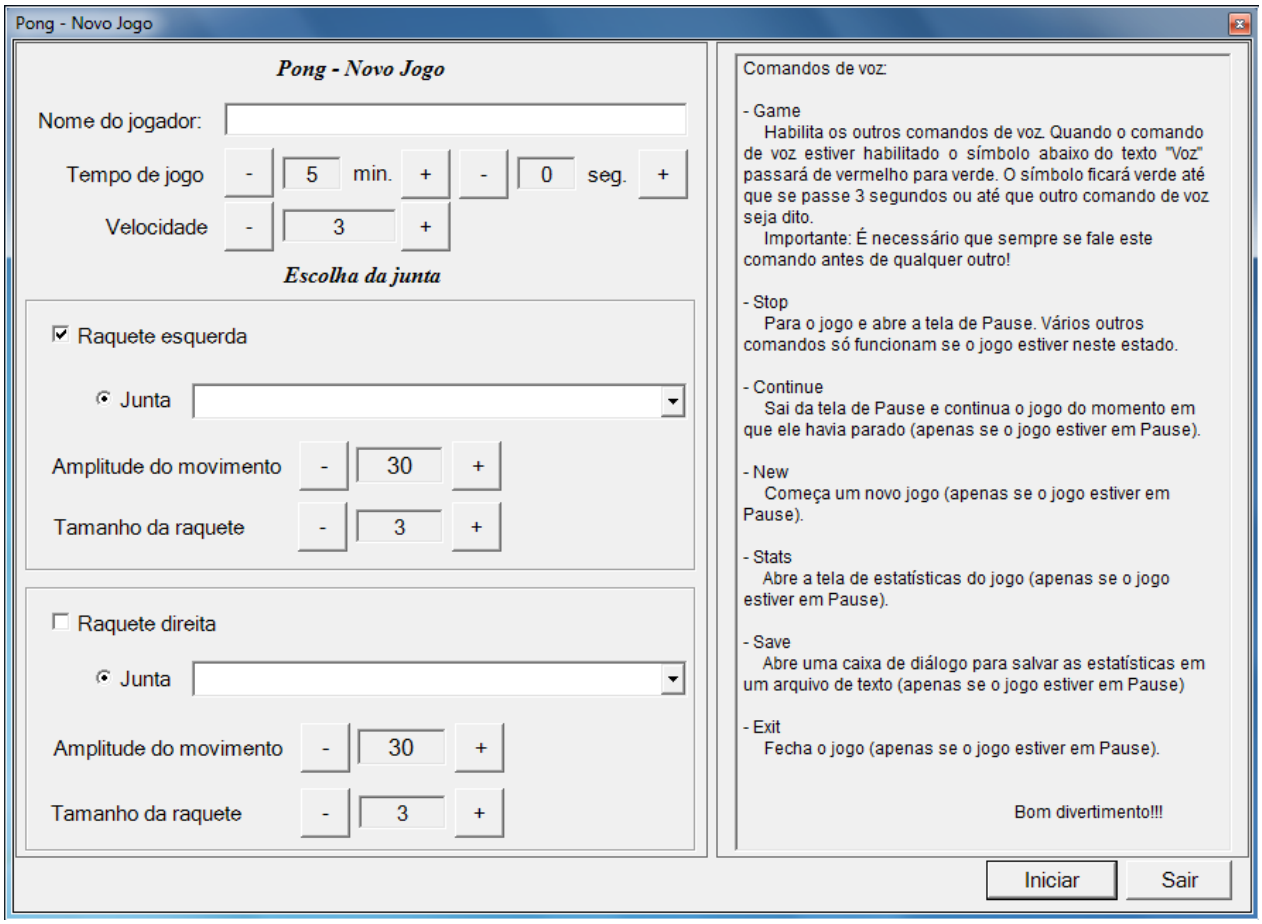


Figura 24 – Tela de Novo Jogo

- Nome do jogador: campo onde pode ser inserido o nome do paciente (opcional);
- Tempo de jogo: indica o tempo total que deve ter o jogo. Separado nos campos dos minutos e dos segundos. Os minutos podem ser ajustados de um em um minuto e os segundos podem der ajustados de dez em dez segundos. Os valores máximos são de noventa e nove para os minutos e cinquenta para os segundos;
- Velocidade: indica a velocidade da bola, podendo variar entre um e cinco, sendo um a velocidade mais lenta e cinco a velocidade mais rápida. Estes valores variam de 0,5 em 0,5;
- Comandos de voz: possui instruções de como utilizar os comandos de voz.

Para cada raquete, esquerda e direita, além de poder indicar se a raquete está ou não habilitada, pode-se configurar:

- Junta: escolhe-se para qual junta deve-se associar o movimento da raquete. As juntas disponíveis são: mãos, joelhos e pés;
- Amplitude do movimento: indica a amplitude máxima do movimento vertical que a junta escolhida pode realizar. Os valores disponíveis variam entre dez e duzentos centímetros, variando de dez em dez centímetros;
- Tamanho da raquete: indica o tamanho da raquete. Os valores possíveis estão entre um (menor raquete possível) e cinco (maior raquete possível), variando de 0,5 em 0,5. Na Figura 25 tem-se um exemplo de diferentes tamanhos de raquetes.

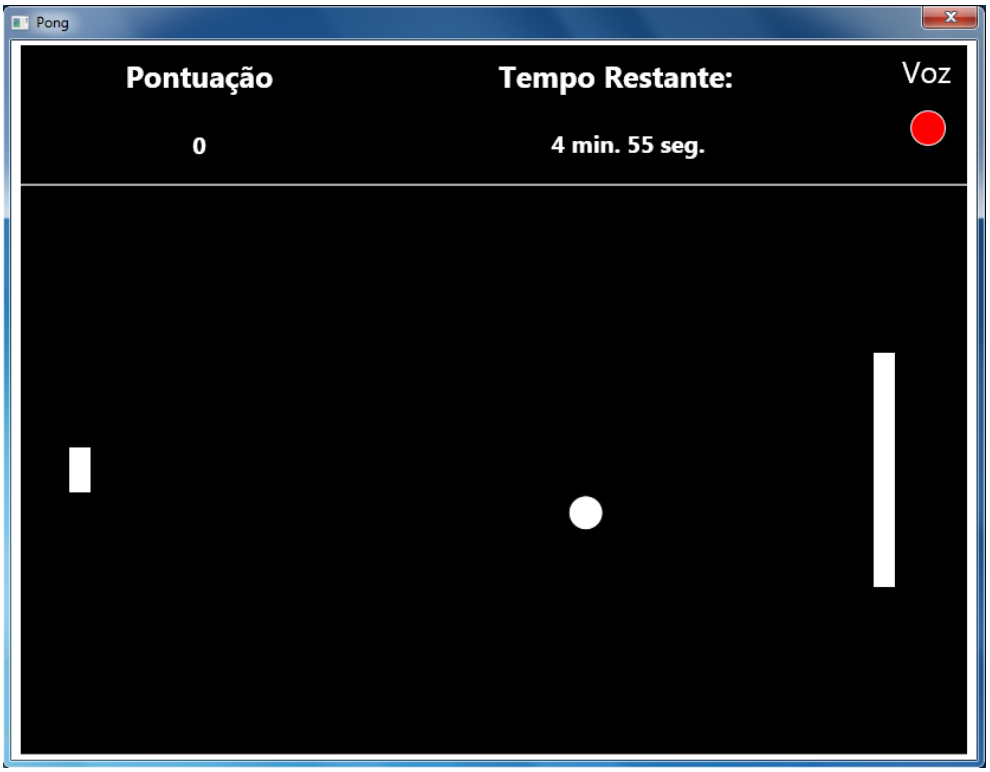


Figura 25 – Raquete esquerda com tamanho 1 e raquete direita com tamanho 5

Na tela de Novo Jogo têm-se os seguintes botões:

- +: incrementa o respectivo atributo;
- -: decrementa o respectivo atributo;

- Iniciar: inicia o jogo com as configurações escolhidas. O botão não fará nada enquanto nenhuma junta tiver sido escolhida ou nenhuma raquete estiver selecionada. Como os outros campos possuem um valor inicial pré-determinado eles nunca ficarão vazios;
- Sair: encerra o aplicativo e fecha todas as janelas.

Ao se clicar no botão Iniciar, a tela do jogo Pong deve ser carregada.

Na Figura 26 a raquete da esquerda está associada a uma junta, porém a raquete da direita não foi associada a nenhuma junta, por isso seu tamanho foi definido como a altura do campo de jogo. Isso ocorre para que o paciente possa jogar com apenas uma raquete.

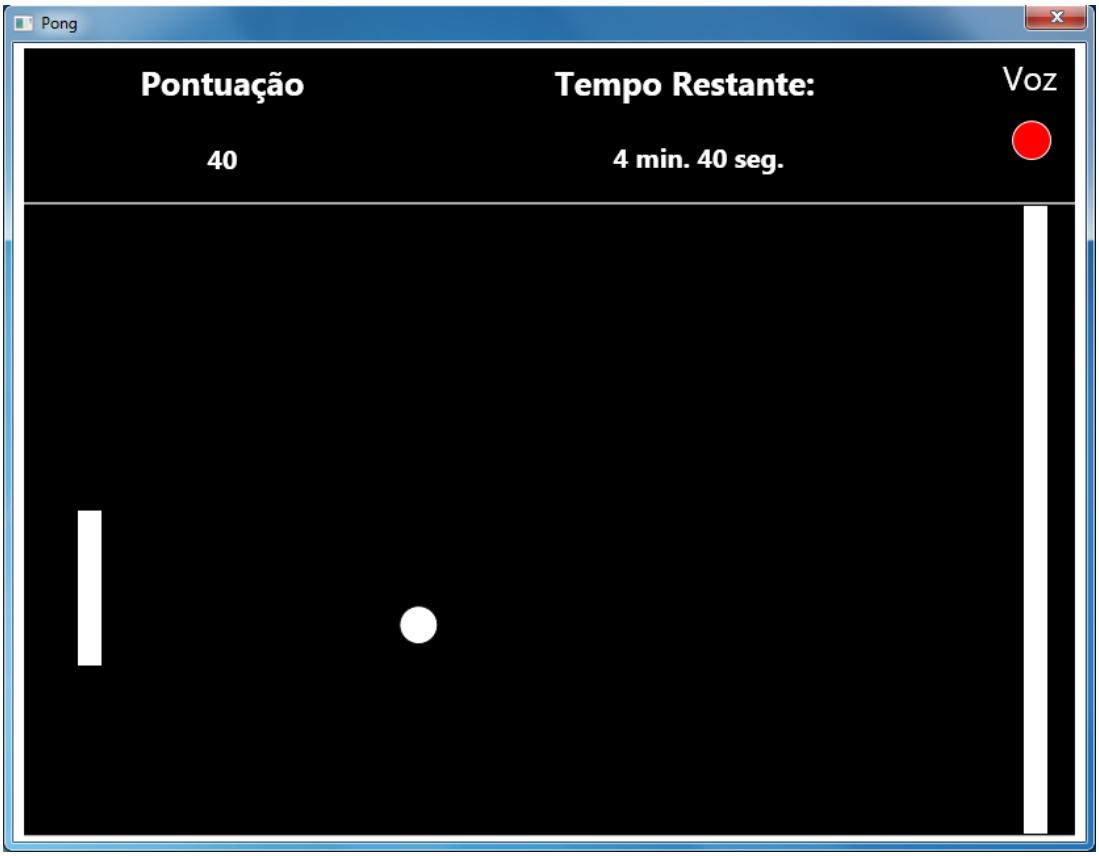


Figura 26 – Tela do jogo Pong com 1 raquete habilitada

A pontuação mostra quantos pontos o paciente obteve até o momento. A pontuação é igual aos acertos no centro das raquetes vezes 30, mais os acertos nas outras partes da raquete vezes 10, menos as bolas perdidas vezes 5.

O tempo restante mostra quanto tempo de jogo ainda resta. Quando o contador chega a zero, o jogo para automaticamente e mostra a tela de fim de jogo (Figura 27) com as estatísticas.

Na parte “Informações do jogo” há os campos:

- Pontuação final: mostra a pontuação obtida ao final do jogo ou a pontuação atual do momento em que o jogo estiver no estado de *Pause*;
- Tempo de jogo: mostra o tempo decorrido do jogo. Será igual ao tempo total escolhido pelo jogador ao final do jogo ou se estiver em estado de *Pause* mostrará o tempo decorrido desde o começo do jogo.

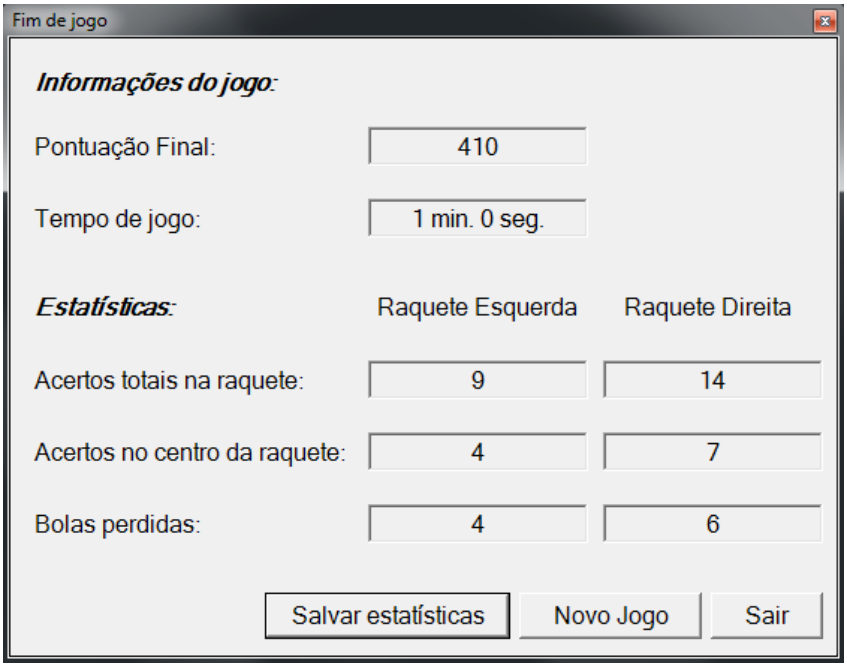


Figura 27 – Tela de fim de jogo

Em “Estatísticas” para cada item há dois campos, um para a raquete esquerda e outro para a raquete direita:

- Acertos totais na raquete: mostra quantas vezes o jogador conseguiu rebater a bola com a raquete;
- Acertos no centro da raquete: mostra quantas vezes o jogador conseguiu rebater a bola usando a parte central da raquete. A parte central da raquete é definida pegando-se a linha de centro da raquete na horizontal e contando-se 20% da altura para cima e 20% para baixo;
- Bolas perdidas: quantas bolas passaram pela raquete e tocaram a borda lateral do campo.

No canto inferior da tela, têm-se os botões:

- Salvar estatísticas: abre a caixa de diálogo (Figura 28) para salvar as estatísticas em um arquivo de texto. Após salvar o arquivo ele é aberto para possibilitar que o usuário veja o seu conteúdo.
- Novo Jogo: deve reiniciar o jogo e mostrar a tela de Novo Jogo (Figura 24).
- Sair: deve fechar todas as janelas e parar os sensores do Kinect.

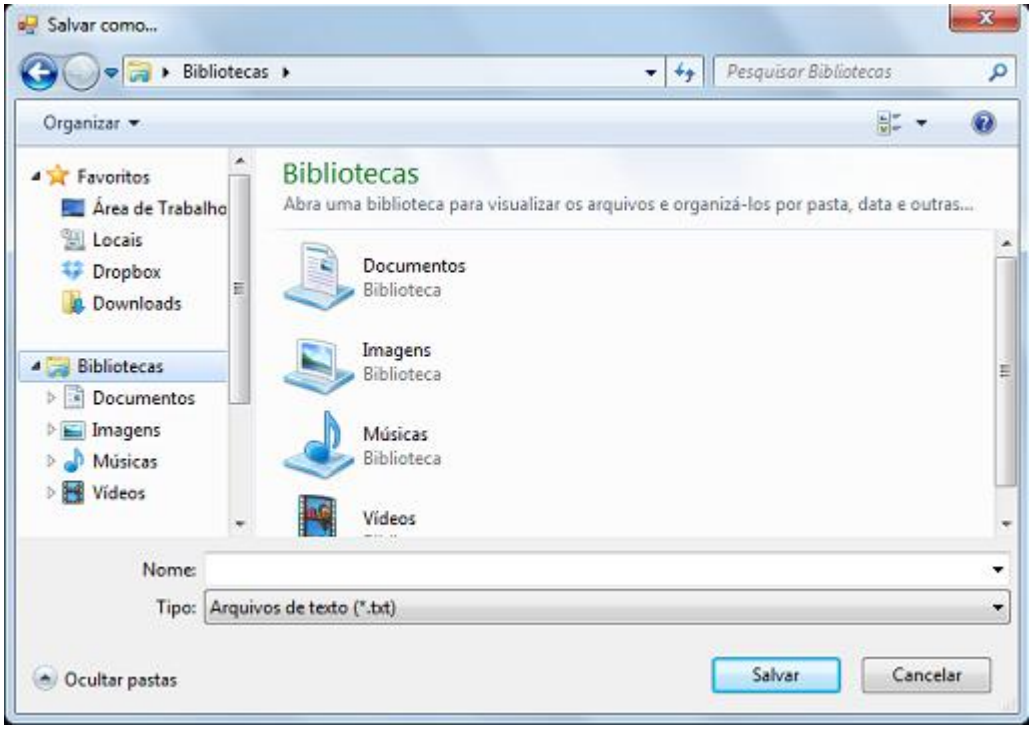


Figura 28 – Caixa de diálogo para salvar o arquivo de texto

Ao se clicar em salvar estatísticas deve-se abrir a caixa de diálogo para salvar o arquivo e logo em seguida ele deve ser aberto mostrando o conteúdo igual ao da Figura 29.

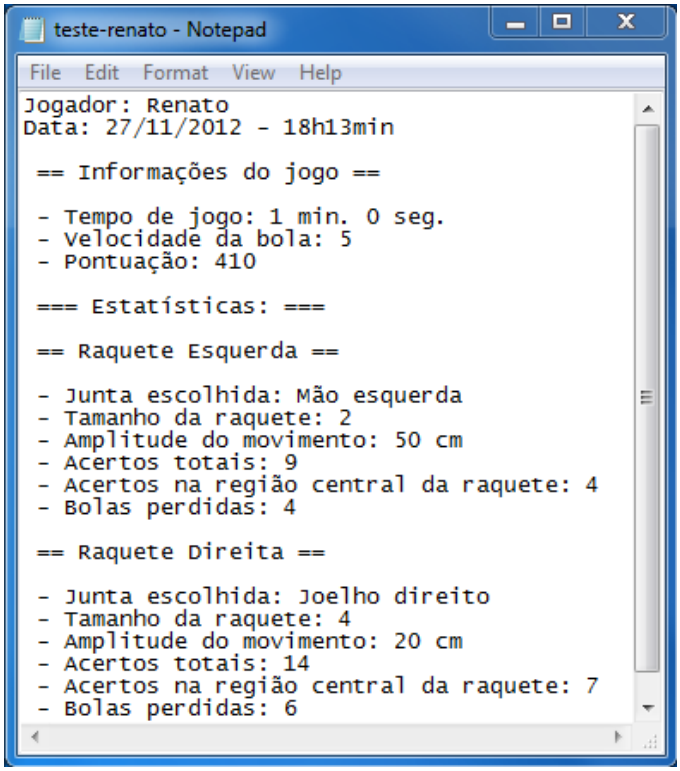


Figura 29 – Arquivo de texto criado contendo as estatísticas do jogo

Para abrir outras telas é necessário usar o comando de voz do Kinect. Para ativar o comando de voz basta dizer a palavra *Game*, isso fará com que os outros comandos de voz sejam habilitados e o ícone de Voz no canto superior direito passe de vermelho para verde (Figura 30).

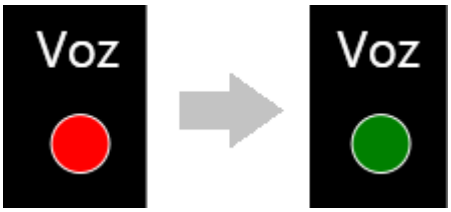


Figura 30 – Comando de voz ativo

O comando de voz ficará ativo até que outro comando seja dito ou até que se passem três segundos sem que seja dito algum comando.

Os possíveis comandos de voz, quando este estiver habilitado, são:

- *Stop*: para o jogo e abre a tela de *Pause* (Figura 31);
- *Continue*: fecha a tela de *Pause* e continua o jogo do ponto em que este havia parado;
- *New*: começa um novo jogo;
- *Stats*: mostra a tela com as estatísticas atuais do jogo;
- *Save*: abre uma caixa de diálogo para salvar as estatísticas do jogo em um arquivo de texto;
- *Exit*: fecha todas as janelas e para os sensores do Kinect.

Os comandos *Continue*, *New*, *Stats*, *Exit* e *Save* só podem ser acionados quando o jogo estiver em modo de *Pause*.

Quando o comando *Stop* for dito, a tela de *Pause* deve ser exibida.

Além dos comandos de voz, pode-se usar os botões da tela para realizar as ações desejadas, com exceção da ação de pausar.

Figura 31 – Tela de *Pause*

O botão “Continuar” possui a mesma função do comando de voz *Continue*, ou seja, deve fazer com que o jogador volte ao jogo.

O botão “Novo Jogo” é igual ao comando *New* e deve reiniciar o jogo.

O botão “Obter Estatísticas” abre a tela que contém as estatísticas (Figura 27) e seu equivalente seria o comando *Stats*.

O botão “Sair” tem a mesma função de *Exit* e deve sair do jogo.

O botão que é análogo ao comando de voz *Save* é o botão “Salvar Estatísticas” da tela Fim de Jogo (Figura 27).

Foi verificado que a velocidade da bola varia muito com a velocidade do processador do computador utilizado. Apesar de a bola possuir uma velocidade máxima, ela não possui uma velocidade mínima. Isso ocorre porque é definida uma taxa mínima de atualização da tela e não há uma taxa máxima de atualização, pois essa taxa máxima poderia acarretar numa perda muito significativa de quadros por segundo durante o jogo, atrapalhando o andamento do programa.

5 CONCLUSÕES

Ao final do projeto, pode-se concluir que foi possível desenvolver um *software* para reabilitação utilizando o Kinect que pode ser instalado facilmente e que pode ser usado na própria casa do paciente, já que ele não necessitará de mais nada além de um PC com Windows 7 ou superior, do programa Kinect for Windows SDK v1.5 e do próprio sensor Kinect para utilizar o programa. O jogo foi desenvolvido com uma interface simples de forma que o paciente e seu auxiliar pudessem jogar em casa facilmente sem a supervisão do médico ou do fisioterapeuta, como em alguns estudos em que o paciente realiza a sessão e mostra apenas os resultados obtidos ao médico, que deve decidir se os parâmetros da sessão devem ser mudados ou não (SCHÖNAUER et al., 2011). Os comandos de voz também contribuem de modo que o paciente não precise sair de sua posição para executar os comandos. Além disso, os testes realizados foram condizentes com o que foi proposto, mostrando que o *software* consegue captar os movimentos de até duas juntas e associá-las ao movimento das raquetes de maneira eficaz, possui níveis de dificuldade adequados a uma sessão de reabilitação, possui parâmetros ajustáveis que permite que o *software* se adapte a cada paciente e que também é motivador para o processo, de modo que o jogo poderá ser utilizado de forma a trazer resultados mais favoráveis à sua reabilitação, já que existem estudos que comprovam que a fisioterapia utilizando o Kinect é até mais eficaz que os métodos convencionais (CHANG, CHEN, HUANG, 2011).

Observa-se que alguns aspectos do projeto ainda podem ser melhorados em futuras implementações. Poderia ser implementado um modo para dois jogadores, para o paciente poder jogar junto com algum amigo ou familiar. Também poderiam ser incluídas as juntas dos cotovelos nas opções, pois os testes realizados mostraram que quando um paciente que não possui parte do braço utiliza o *software*, escolhendo a mão como junta, o Kinect pode não reconhecê-la. Outro aspecto seria incluir um botão de pausa na tela, pois, apesar de pequena, existe a possibilidade de tanto o paciente quanto o auxiliar estarem impossibilitados de executar os comandos de voz, além de ser mais rápido e prático para o auxiliar clicar no botão para pausar. Além disso, também seria possível trabalhar em conjunto com um médico fisioterapeuta no aprimoramento do *software* para incluir outros fatores que possam contribuir com a

sessão de reabilitação como, por exemplo, a opção de gravar vídeos do paciente. Caso seja lançada uma versão do SDK do Kinect que possua o pacote de idiomas em português do sistema de reconhecimento de voz, também seria possível traduzir os comandos de voz.

Com o conhecimento adquirido no desenvolvimento do código é possível criar novos programas com outros jogos para reabilitação ou até mesmo programas para o uso cotidiano.

6 REFERÊNCIAS BIBLIOGRÁFICAS

Bó, A. P. L.; Hayashibe, M.; Poignet, P. *Joint Angle Estimation in Rehabilitation with Inertial Sensors and its Integration with Kinect*, In. **33rd Annual International Conference of the IEEE EMBS**, Boston, Massachusetts USA, pp. 3479-3483, 2011.

Booch, G.; Rumbaugh, J.; Jacobson, I. ***The Unified Language User Guide***, 1998. Publisher: Addison Wesley.

Chang, Y. C.; Chen, S. F.; Huang, J. D. *A Kinect-based for physical rehabilitation: A pilot study for young adults with motor disabilities*. In. **Research in Developmental Disabilities**, Vol. 32, No. 6, pp. 2566-2570, 2011.

Gama, A.; Chaves, T.; Figueiredo, L.; Teichrieb, V. *Improving Motor Rehabilitation Process through a Natural Interaction Based System Using Kinect Sensor*. In. **IEEE Symposium on 3D User Interfaces**. Orange Country, CA, USA, pp. 145-146, 2012.

Gotsis, M.; Lympouridis, V.; Turpin, D.; Tasse, A.; Poulos, I. C.; Tucker, D.; Swider, M.; Thin, A. G.; Jordan-Mash, M. *Mixed Reality Game Prototypes for Upper Body Exercise and Rehabilitation*. In. **IEEE Virtual Reality**. Orange Country, CA, USA, pp. 181-182, 2012.

Hasenbring, M. *Attentional control of pain and the process of chronification*. In. **Progress in brain research**, Vol. 29, pp. 525-534, 2000.

Lange, B.; Chang, C. Y.; Suma, E.; Newman, B.; Rizzo, A. S.; Bolas, M. *Development and Evaluation of Low Cost Game-Based Balance Rehabilitation Tool Using the Microsoft Kinect Sensor*. In. **33rd Annual International Conference of the IEEE EMBS**. Boston, Massachusetts USA, pp. 1831-1834, 2011.

Leyvand, T.; Meekhof, C.; Wei, Y. C.; Sun, J.; Guo, B. *Kinect Identity: Technology and Experience*. In. **Computer**, Vol. 44, No. 4, pp. 94-96, 2011.

Microsoft. ***Kinect for Windows SDK Documentation***, 2012. Disponível em <http://msdn.microsoft.com/en-us/library/hh855347.aspx>

Microsoft. ***Programming with the Kinect for Windows SDK***, 2011. Disponível em: <http://research.microsoft.com/en-us/events/fs2011/jancke_kinect_programming.pdf>.

Paula, B. C. Adaptando jogos para uso com o Microsoft Kinect. In. **Proceedings of SBGames 2011**, Salvador, pp. 1-13, 2011.

Schönauer, C.; Pintaric, T.; Kaufmann, H.; Jansen-Kosterink, S.; Vollenbroek-Hutten, M. *Chronic Pain Rehabilitation with a Serious Game using Multimodal Input*. In. **Proceedings of the International Conference on Virtual Rehabilitation 2011**, Switzerland, pp. 1-8, 2011.

APÊNDICE A - UML

As informações usadas na descrição da UML foram retiradas de Booch et al. (1998).

Para compreender o que é apresentado no projeto, é necessário possuir um conhecimento básico em UML. Levando isso em conta, é feita uma breve descrição de cada diagrama UML que é utilizado no projeto.

A *Unified Modeling Language* (UML) permite uma melhor visualização da estrutura do projeto de *softwares* por meio de diversos diagramas. A UML pode ser utilizada para visualizar, especificar, construir e documentar o sistema de um *software*.

O diagrama é uma representação gráfica de um conjunto de elementos, mais comumente representado como um grafo de vértices (objetos) e arcos (relacionamentos). Na teoria, um diagrama pode conter qualquer combinação de objetos e relacionamentos. Na prática, entretanto, surge um pequeno número de combinações comuns que são consistentes com as 5 visões mais úteis que compreendem a arquitetura de um sistema de um *software*. Por esta razão, a UML inclui quatorze diagramas:

1. Diagrama de classe
2. Diagrama de objetos
3. Diagrama de casos de uso
4. Diagrama de sequência
5. Diagrama de colaboração
6. Diagrama de estados
7. Diagrama de atividade
8. Diagrama de componentes
9. Diagrama de instalação
10. Diagrama de pacotes
11. Diagrama de estrutura
12. Diagrama de interatividade
13. Diagrama de tempo
14. Diagrama de perfil

Para este projeto se faz necessário a representação de somente 4 deles: de casos de uso, de atividade, de classe e de sequência. Estes 4 diagramas serão descritos a seguir.

A.1 Diagrama de casos de uso

Os diagramas de casos de uso são centrais para modelar o comportamento de um sistema, de um subsistema, ou de uma classe. Cada um mostra um conjunto de casos de uso e atores e seus relacionamentos. Os diagramas de casos de uso são importantes para visualizar, especificar e documentar o comportamento de um elemento para que os usuários compreendam como usá-lo e para que os desenvolvedores possam implementá-lo. Eles tornam os sistemas, subsistemas e classes acessíveis e compreensíveis apresentando uma visão externa de como esses elementos podem ser usados no contexto. Diagramas de casos de uso também são importantes para testar sistemas executáveis através de engenharia direta e compreendê-los através de engenharia reversa.

Um diagrama de casos de uso normalmente consiste de:

- Casos de uso
- Atores
- Dependências, generalizações e relacionamentos de associação

Como todos os outros diagramas, este também pode conter notas e restrições.

Os elementos básicos do diagrama de casos de uso (Figura 32) são o ator, a fronteira do sistema, o caso de uso, a relação de associação e a relação de dependência.

O ator, representado por um boneco, representa um usuário ou qualquer outra entrada externa que interage com o sistema.

A fronteira do sistema delimita quais componentes farão parte do sistema e quais serão externos. Atores são representados fora da caixa retangular que delimita a fronteira do sistema. Dentro da caixa estão contidas as diferentes funções ou casos de uso do sistema.

Casos de uso indicam as ações executadas por um sistema, que possuem um resultado observável. Cada caso de uso representa uma porção concisa de uma funcionalidade e irão interagir de alguma maneira com os atores. Os casos de uso são representados por uma elipse com o nome do caso de uso dentro ou abaixo dela.

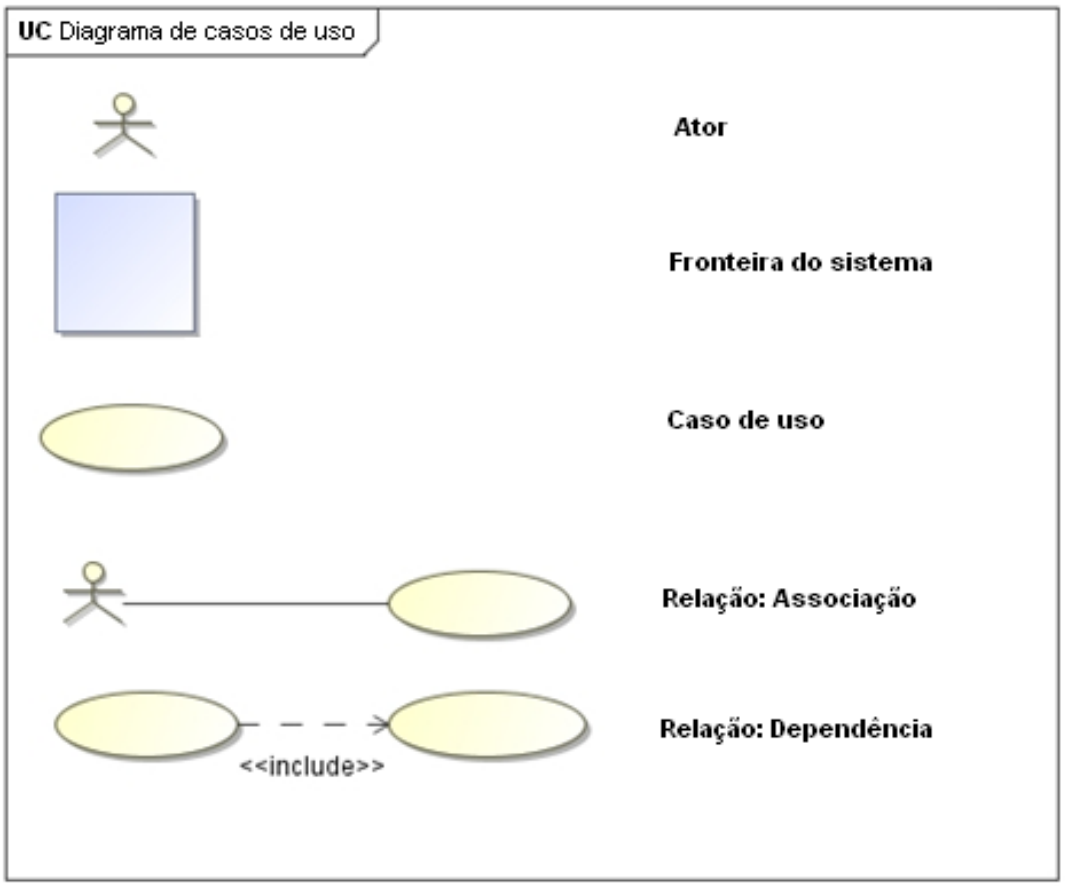


Figura 32 – Principais elementos do diagrama de casos de uso

A relação de associação se refere a como os casos de uso podem ser interligados baseados numa certa funcionalidade. A relação é representada como um segmento de reta ou uma seta do elemento original até o caso de uso apropriado.

A relação de dependência, também conhecida como relação de inclusão, é uma relação direta entre dois casos de uso, implicando que o comportamento do caso de uso incluído é inserido no comportamento do caso de uso inclusor. Esta relação indica uma obrigatoriedade do caso de uso incluir a funcionalidade do caso de uso incluído.

Assim, sempre que o primeiro ocorrer, obrigatoriamente o incluído ocorrerá. A notação é uma seta tracejada do caso de uso inclusor para o incluído com o estereótipo <<include>>.

A.2 Diagrama de atividade

Um diagrama de atividades é essencialmente um fluxograma, mostrando o fluxo de controle de atividade para atividade. Usam-se diagramas de atividade para modelar os aspectos dinâmicos de um sistema. Na maior parte, isso envolve modelar os passos sequenciais num processo computacional. Com um diagrama de atividades pode-se também modelar o fluxo de um objeto quando ele se move de estado para estado em pontos diferentes no fluxo de controle. Diagramas de atividades também podem ser utilizados para visualizar, especificar, construir e documentar a dinâmica de uma sociedade de objetos ou utilizados para modelar o fluxo de controle de uma operação.

Os elementos básicos do diagrama de atividades, que podem ser visualizados na Figura 33, são:

- Início;
- Fim;
- Ação – representa alguma transformação que ocorre no sistema;
- Fluxo entre ações – representa a transição entre uma ação e outra;
- Bifurcação – separa uma transição em várias executadas ao mesmo tempo;
- Sincronização – combina duas ou mais transições em uma;
- Decisão – mostra as diferentes transições dependendo de uma condição;
- Evento de entrada – ocorre a recepção de um sinal;
- Envio de um evento – ocorre o envio de um sinal para um meio externo, por exemplo, um *hardware*.

A.3 Diagrama de classe

Os diagramas de classe são os diagramas mais comumente encontrados em modelagem de sistemas orientados a objetos. Eles mostram um conjunto de classes, interfaces, colaborações e seus relacionamentos. Usa-se o diagrama de classe para modelar a visão estática do projeto de um sistema. Diagramas de classe são

importantes não só para visualizar, especificar e documentar modelos estruturais, mas também para construir sistemas executáveis por meio de engenharia direta e reversa.

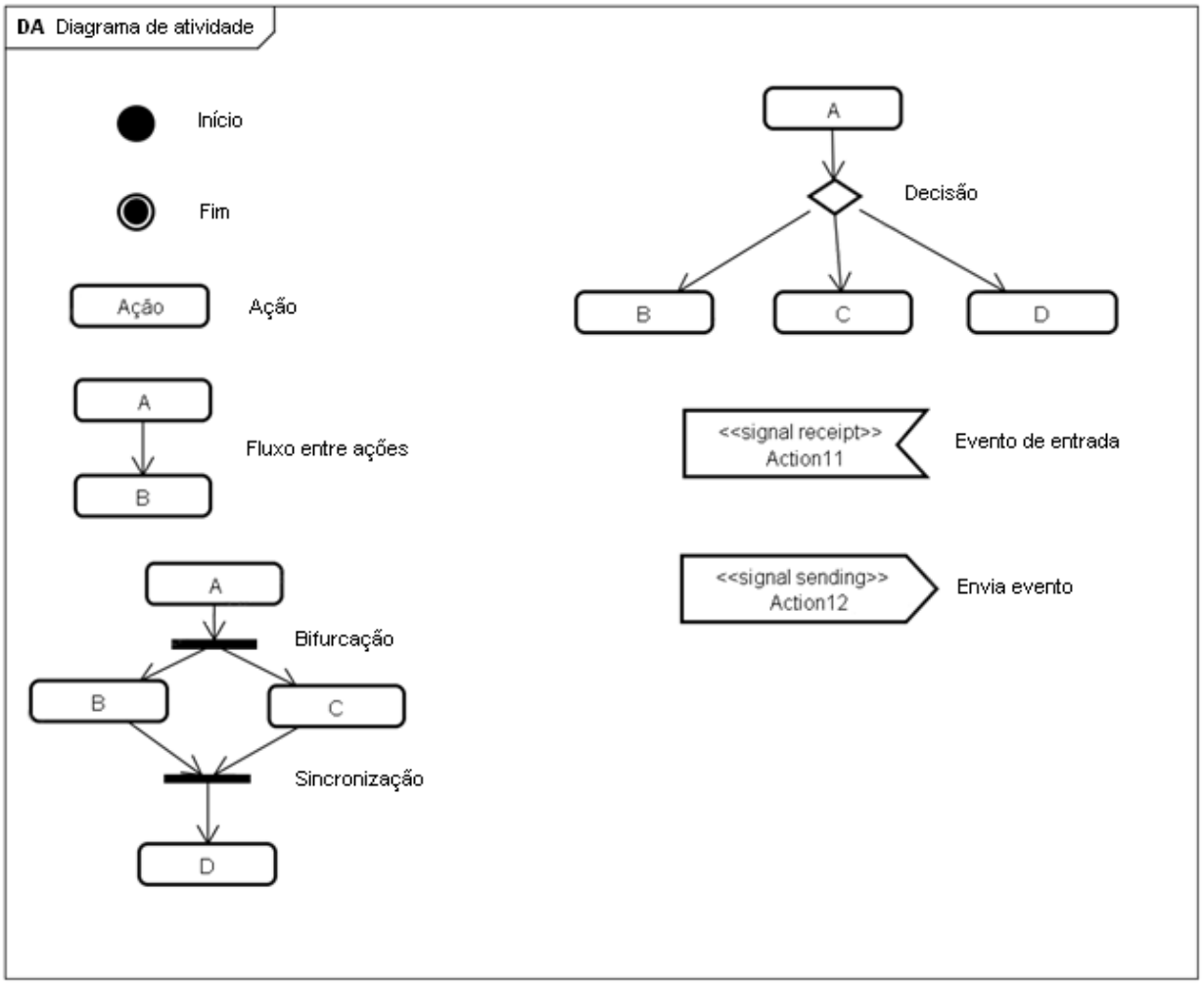


Figura 33 – Principais elementos do diagrama de atividades

Os elementos básicos do diagrama de classe (Figura 34) são a classe, a herança e a relação de associação (que inclui a de agregação e de composição).

A classe é um elemento abstrato que representa um conjunto de objetos. A classe contém a especificação do objeto, suas características, atributos e métodos.

A herança é um princípio que permite que classes compartilhem atributos e métodos. Ela é usada na intenção de reaproveitar código ou comportamento

generalizado ou especializar operações ou atributos. No exemplo da Figura 34, a classe “Filho” herda os atributos e métodos da classe “Pai”.

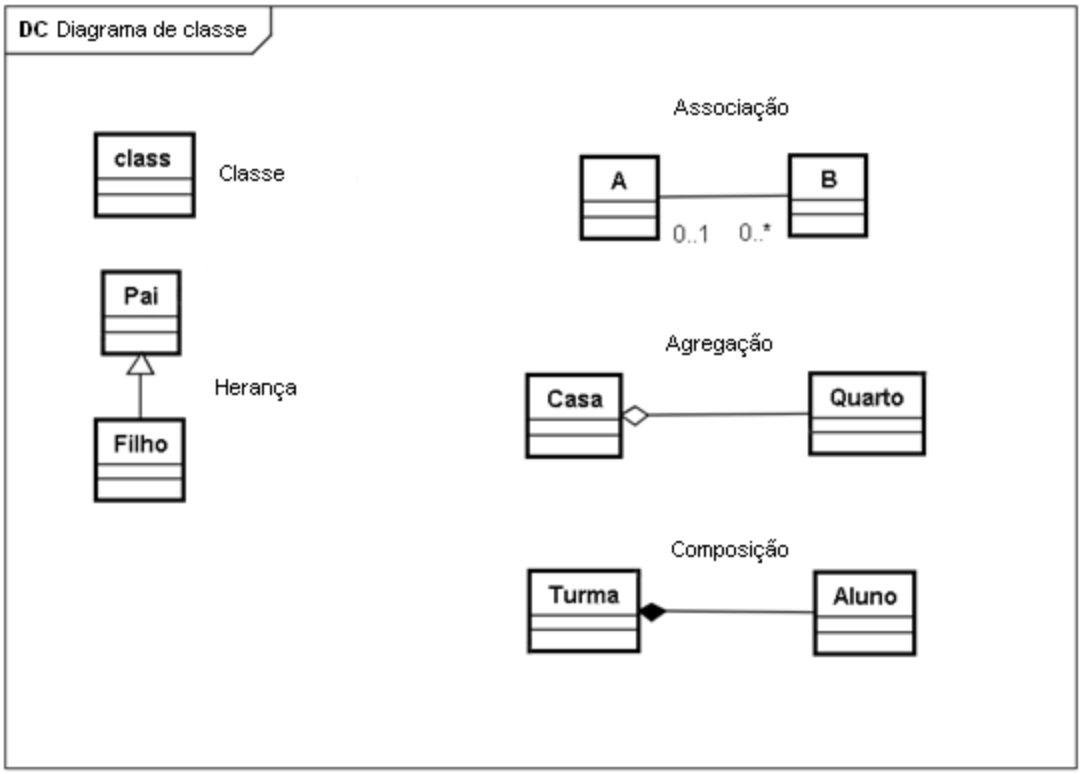


Figura 34 – Principais elementos do diagrama de classes

A relação de associação representa o relacionamento entre classes. Existem quatro tipos de associação: bidirecional, unidirecional, agregação (que inclui composição) e reflexiva. Serão discutidas as relações de agregação e composição somente, pois são as relações relevantes para o projeto.

A relação de agregação é uma associação mais específica que representa um relacionamento entre uma parte e um todo. No exemplo da Figura 34, a classe “Quarto” é parte do todo, que é a classe “Casa”.

A relação de composição é uma agregação mais específica, que representa um relacionamento de contenção. Neste caso, um elemento contém outro elemento e esses elementos que estão contidos dentro de outro dependem dele para existir. Se um

elemento é destruído, os elementos que estão contidos nele também o são. No exemplo da Figura 34, a classe “Turma” contém a classe “Alunos”.

A.4 Diagrama de sequência

Um diagrama de sequência é um diagrama de interação que enfatiza a ordem das mensagens. Um diagrama de sequência mostra um conjunto de objetos e as mensagens enviadas e recebidas por eles. Esses objetos são tipicamente instâncias nomeadas ou anônimas de classes, mas também podem representar instâncias de outras coisas como colaborações, componentes e nós. Usa-se o diagrama de sequência para ilustrar a visão dinâmica de um sistema. Cada diagrama representa um caso de uso e esses diagramas, de uma forma geral, são análogos aos diagramas de atividade, porém com mais riqueza de detalhes visto que estes especificam as classes e os respectivos métodos responsáveis por executar uma tarefa.

Os elementos básicos do diagrama de sequência (Figura 35) são o ator, o objeto, a mensagem, o fragmento e a linha de vida.

O ator é uma entidade externa que interage com o sistema e que solicita serviços, gerando dessa forma eventos que iniciam processos.

Os objetos representam as instâncias das classes representadas no processo. Os objetos são ilustrados como retângulos. Eles compõem a dimensão horizontal.

A mensagem representa a interação entre os objetos, ela contém a assinatura do método que está sendo chamado. A mensagem pode ser para outro objeto ou para o mesmo objeto (auto-mensagem). Ela também pode ser síncrona, representada por uma seta cheia, ou assíncrona, representada por uma seta vazia. A seta tracejada representa uma mensagem de retorno.

Um fragmento de interação pode ser do tipo: *Alt* (Alternativa), *Opt* (Opcional), *Break* (Parar), *Loop* (Repetição), entre outros.

As linhas de vida compõem a dimensão vertical (tempo). A dimensão vertical é a sequência onde é representada a vida do objeto durante a interação.

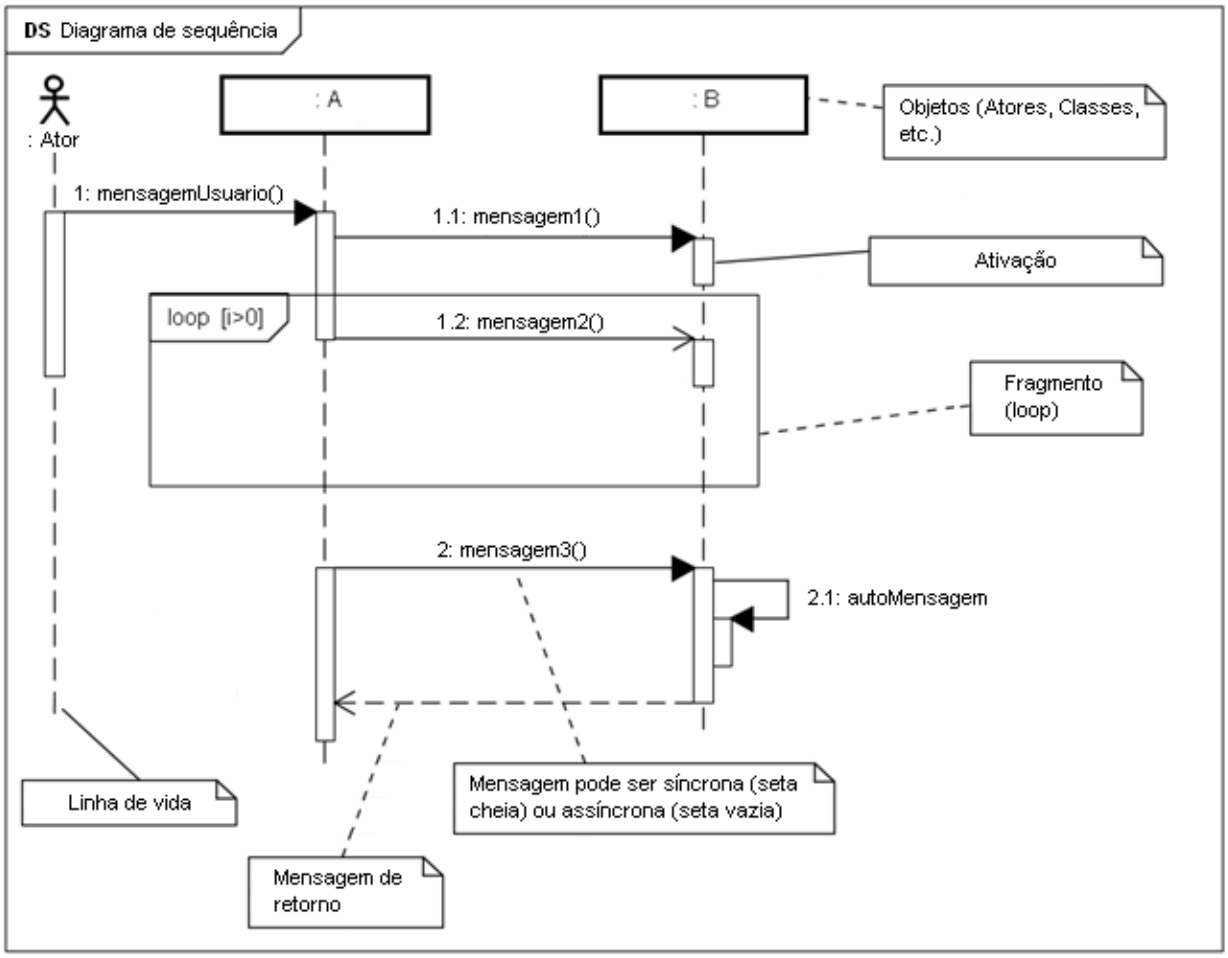


Figura 35 – Principais elementos do diagrama de sequência