IRTaktiks: Jogo de Estratégia para Mesa Multitoque

Willians S. Schneider Nilson C. Dias F.º Luis H. M. Mauruto Fábio R. Miranda

Centro Universitário Senac, Bacharelado em Ciência da Computação
São Paulo – SP, Brasil

Figura 1: *IRTaktiks* jogado simultaneamente por dois jogadores, que manipulam suas unidades diretamente com as mãos

Abstract

We present *IRTaktiks*, a real-time two-player strategy game playable on a custom multi-touch table.
Details regarding the construction of the interface hardware, the detection and recognition of events and the architecture of the software that was designed and implemented are presented. Aspects of how the game works and results of the finished project are also presented and discussed.

Keywords: Architectures, Engines, and Design Patterns; Computer Graphics, Human-computer Interfaces, Interface hardware, Programming Techniques.

Authors' contact:
willians.schneider@gmail.com,{lhmenezes, nilmesmo}@gmail.com,
fabio.rmiranda@sp.senac.br

Sample Video:
www.tinyurl.com/irtaktiks/

Source code:
http://www.codeplex.com/irtaktiks

1. Introdução

Em tempos recentes tem havido interesse crescente em dispositivos de interação multitoque, caracterizados por um ou mais usuários poderem interagir através dos dedos diretamente com uma interface gráfica, dispensando o uso de dispositivos de entrada mais convencionais, como teclados, mouses ou canetas especiais. Inicialmente protótipos de telas e mesas de interação multitoque desenvolvidos por pesquisadores de interação e entusiastas foram demonstrados, e agora este conceito já se encontra embutido em produtos computacionais, o telefone celular *iPhone* ou no futuro *Windows* 7.

Uma materialização bastante comum das interfaces multitoque é na forma de mesas, que são interessantes por ser um objeto do domínio cotidiano das pessoas e naturalmente um meio para colaboração, quer na forma de espaço de trabalho, quer na forma de jogos. As interfaces multitoque podem trazer para o ambiente computacional uma forma de colaboração e interação simultânea que normalmente é inviabilizada nas estações de trabalho convencionais por questões ergonômicas (estão disponíveis apenas um mouse e teclado), mas que muitas vezes está presente em jogos de mesa e tabuleiro.

Tanto na forma de mesas de interação quanto na de outros tipos de dispositivos, é razoável supor que o potencial de interação intuitiva dos dispositivos de interação multitoque aliado ao suporte crescente ao desenvolvimento de aplicações deste tipo em sistemas operacionais e em bibliotecas independentes constitui uma oportunidade interessante para o desenvolvimento de jogos inovadores.

Este trabalho trata do projeto e implementação do *IRTaktiks*, um jogo de estratégia jogável simultaneamente por dois jogadores (exemplificado na Figura 1) numa mesa de interação multitoque desenvolvida com materiais de baixo custo (que pode ser vista na Figura 5). Serão apresentados aspectos relacionados à construção do dispositivo de interação, detecção dos múltiplos toques a partir do uso de bibliotecas de código aberto, transformação dos toques em eventos de jogo e o significado destes eventos no domínio do jogo. Este trabalho também detalha elementos de projeto do jogo.

A organização deste artigo é descrita a seguir. Na próxima seção alguns trabalhos relacionados que tratam de interação multitoque são apresentados. Na seção 3 a reflexão total interna frustrada da luz (*FTIR*), que é o princípio do dispositivo de entrada usado neste trabalho, é apresentada. A infra-estrutura de software usada no projeto é discutida na seção 4 enquanto na seção 5 apresentam-se detalhes do desenvolvimento do projeto, cujos resultados são apresentados na seção 6. A seção 7 trata das conclusões e trabalhos futuros.

2. Trabalhos relacionados

A interação multitoque popularizou-se em 2006 com a ajuda de vídeos na Internet em que *Jefferson Y. Han*, pesquisador do instituto de ciências matemáticas *Courant* demonstrou seu trabalho de pesquisa de interação multitoque utilizando uma superfície com display gráfico interativa que permitia a interação simultânea de múltiplos usuários. Foram apresentadas implementações de diversas aplicações, entre as quais jogos multitoque simples. Os protótipos de *J. Han* [Han 2005] despertaram o interesse de diversas iniciativas de pesquisa sobre essa nova alternativa de interação, e logos estavam disponíveis na WWW diversos tutoriais e weblogs [Buxton 2008] que trocam experiências entre pesquisadores e fomentam o desenvolvimento de protótipos.

O multitoque teve seu início em 1982 [Multigesture.net 2008], com tablets feitos na universidade de *Toronto* e com telas dos laboratórios *Bell*. Nos anos 90 a universidade de *Delaware* desenvolveu um sofisticado sistema de reconhecimento de gestos e escrita, base para o mouse-pad *iGesture* e teclados *TouchStream*, comercializados pela *FingerWorks* em 2001. Estes teclados eram reconhecidos pela sua ergonomia, os usuários apenas precisavam apontar e arrastar com um ou mais dedos, eliminando totalmente a necessidade de um dispositivo apontador, como o mouse.

O primeiro dispositivo multitoque com display integrado a ser comercializado foi o *Lemur Input Device*, um controlador multimídia profissional da companhia francesa *JazzMutant* lançado em 2005 [Multigesture.net 2008]. Em julho de 2007, a *Apple Inc*. lançou o produto *iPhone* que tinha interação multitoque e a empresa Microsoft demonstrou logo a seguir uma mesa de interação chamada *Microsoft Surface*.

A mesa *ReacTable* [Kaltenbrunner et al. 2006] funciona como um instrumento musical colaborativo, que permite o reconhecimento de objetos postos sobre sua superfície e tem a possibilidade de permitir interação multiusuário. Na *ReacTable* o usuário pode sintetizar sons através de uma cadeia de fontes, filtros e osciladores manipuláveis, todos gerados por software. Cada objeto colocado sobre sua superfície é classificado por um software a partir de marcadores fiduciais situados em sua superfície e capturados por uma câmera situada abaixo da mesa. Cada objeto é classificado como um dos geradores e filtros de uma aplicação musical obtendo-se como resultado um som único, resultado da interação destes objetos. Este instrumento utiliza como base o software de detecção de fiduciais *ReacTIVision*, que reconhece os objetos sobre a mesa. A *ReacTable* ganhou notoriedade recentemente ao ser utilizado em shows e apresentações pela cantora *Björk*, no *Coachella Festival*, em 2007 na *California* [Wired 2008].

3. FTIR e Multitoque

Atualmente, existem diversas técnicas para a detecção de múltiplos toques em superfícies, por exemplo, análise da imagem de câmeras que enquadram o dispositivo, utilização de sensores medidores de pressão, utilização de um grids de filamentos eletrônicos, onde o toque simplesmente fecha contato permitindo a passagem de corrente elétrica, até a utilização de circuitos eletrônicos que percebem o contato dos dedos humanos por uma alteração de capacitância na região do toque.

As técnicas mais simples e baratas costumam utilizar iluminação infravermelha e uma superfície de acrílico. Em geral um anteparo de projeção é colocado junto ao acrílico, e a superfície de interação é iluminada lateralmente com luz infravermelha. A mesa multitoque utilizada neste trabalho (que pode ser vista na Figura 5) baseia-se em iluminação infravermelha, mais especificamente a *FTIR* (reflexão total interna frustrada da luz).

A reflexão da luz é o fenômeno físico em que um feixe de luz incide sobre uma determinada superfície e é refletida para o mesmo meio de propagação de origem. Quando a reflexão é total, todos os fótons do feixe de luz são redirecionados ao meio de propagação de origem, ao contrário da reflexão parcial, em que alguns fótons atravessam a interface entre os meios de propagação, caso em que ocorre um desvio no ângulo de incidência do feixe de luz emitido, chamado de refração. O pesquisador *J. Y. Han*, durante suas pesquisas sobre técnicas de interação multitoque, percebeu que a pele é um material difusor, ou seja, quando um feixe de luz que seria refletido totalmente entra em contato com a pele, ele é difundido em todas as direções. A esse efeito de difundir a luz que seria totalmente refletida, se deu o nome de reflexão total interna frustrada da luz, que encontra-se ilustrado na Figura 2.

Este fenômeno é passível de aplicação em interfaces multitoque, em que pode-se em iluminar as laterais de uma superfície de acrílico com diversos *LEDs* infravermelhos de modo que a luz emitida fique presa dentro do acrílico devido ao fenômeno da reflexão total da luz. Quando o dedo do usuário toca a superfície da mesa, a luz é difundida para baixo, onde

uma *webcam* obtém imagens. Essa difusão é reconhecida como pontos de alta luminosidade na imagem capturada e a posição dos toques é facilmente detectada. Pode-se ver este fenômeno em efeito na imagem mostrada na Figura 6.

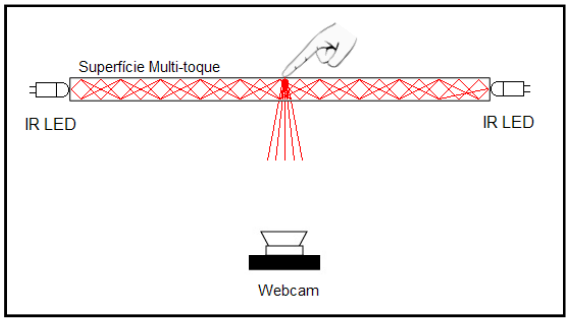

Figura 2: A reflexão total interna da luz na superfície de acrílico é frustrada no ponto de contato com o dedo do usuário

4. Infra-Estrutura e Ferramentas

Alguns softwares publicamente disponíveis foram utilizados para o reconhecimento de toques sobre a superfície da mesa multitoque. Existe uma grande quantidade de superfícies multitoque sendo desenvolvidas por uma comunidade de entusiastas que trocam informações pela Internet, e um padrão para o armazenamento das informações relacionadas aos toques e sua integração com outras aplicações foi sendo adotado pelos desenvolvedores de software.

Durante o projeto da *ReacTable*, desenvolveu-se um protocolo capaz de armazenar informações sobre toques e objetos em qualquer superfície multitoque. A esse protocolo deu-se o nome de *TUIO*.

Outro componente importante do sistema da *ReacTable* é o *reacTIVision*, [Kaltenbrunner et al. 2007] software responsável pela identificação de toques e fiduciais, que analisa a imagem da superfície e emite mensagens do tipo *TUIO* [Kaltenbrunner et al. 2006]. Para comunicar os pacotes *TUIO* com informação sobre os toques com aplicações voltadas a gerar sons e efeitos sonoros, na *ReacTable* tais pacotes são encapsulados num outro protocolo que é compatível com uma grande variedade de bibliotecas de código aberto, chamado *OSC* (*Open Sound Control*)

Após o desenvolvimento do *reacTIVision*, diversos softwares com o propósito de identificação de toques foram desenvolvidos. A grande maioria buscou seguir o mesmo padrão, ou seja, mensagens *TUIO* sob o protocolo *OSC*, tornando-os padrão no desenvolvimento de aplicações multitoque. Um exemplo de projeto que adotou mensagens *TUIO/OSC* é a biblioteca *Touchlib*, adotada no *IRTaktiks* para o reconhecimento dos toques na mesa desenvolvida devido à sua estabilidade e funcionalidades úteis ao projeto. A adoção do *Touchlib* é um fator interessante, pois permite que o projeto *IRTaktiks* funcione corretamente em qualquer superfície multitoque que siga os padrões propostos.

A seguir serão fornecidos mais alguns detalhes a respeito de *TUIO*, *OSC* e *Touchlib*.

4.1 Open Sound Control (OSC) e TUIO

O *Open Sound Control* (*OSC*) [Cnmat 2008] é um protocolo desenvolvido para a comunicação entre computadores, sintetizadores de som e outros dispositivos multimídia. É utilizado em produtos de Realidade Virtual, interfaces Web e também como meio de transporte para outros protocolos que não possuem facilidade de comunicação.

A biblioteca *OSCpack* [Bencina 2008] é um conjunto de classes em *C++* responsáveis por criar e ler pacotes do protocolo *OSC*, incluindo as funcionalidades mínimas para a comunicação utilizando *UDP* nas plataformas *Windows* e *POSIX*. Atualmente é utilizada em diversos projetos, como o *ReacTIVision*, *Touchlib* e *AudioMulch* por causa de sua capacidade de prover comunicação simplificada entre as plataformas *Windows*, *Linux* e *Mac OS X*.

TUIO é um protocolo de comunicação desenvolvido com a finalidade de atender os requisitos de comunicação entre interfaces tangíveis. Este protocolo define propriedades comuns baseadas no controle de objetos, toques e gestos. Há implementações do *TUIO* nas linguagens *Java*, *C*, *C++* e *Adobe Flash*, entre outras.

As mensagens do *TUIO* dividem-se em perfis baseados na interação com a interface tangível. Atualmente, o *TUIO* possui perfis para interfaces 2D, 3D e interfaces customizadas. Cada perfil, por sua vez, possui dois tipos de mensagens diferentes, usadas na representação da interação de objetos e toques com o dispositivo. A mensagem carrega diversas informações sobre a interação, dentre as quais destacam-se a sessão, um identificador da interação, posições no espaço 2D ou 3D, ângulo, vetor de movimento, vetor de rotação, aceleração de movimento e aceleração de rotação. [Tuio 2008]

4.2 Touchlib

O *Touchlib* é uma biblioteca que permite a detecção de toques em superfícies multitoque que utilizam o princípio da reflexão total interna frustrada da luz, quer se utilizem de iluminação traseira ou iluminação frontal. Esta biblioteca foi desenvolvida pela *Natural User Interface Group* em parceria com a *White Noise Audio* [Nuigroup 2008].

Através de algoritmos de divisão e comparação, detecta realces no histograma das imagens enviadas

por uma *webcam* transformando-os em informações sobre cursores, e disparando eventos que podem ser tratados em aplicações *C/C++*. Estes eventos são disparados nos momentos em que um dedo toca, percorre ou é retirado da superfície multitoque. Esta biblioteca permite a integração com demais aplicativos através de *TUIO/OSCpack*.

O *Touchlib* aplica técnicas de processamento de imagem no resultado capturado das *webcams* a fim de melhorar a percepção de toques. Atualmente esta biblioteca trabalha apenas na plataforma *Windows*, porém existem esforços sendo realizados para portá-la para outras plataformas, como *Mac OS X* e *Linux*.

Os eventos detectados pelo *Touchlib* são convertidos para uma escala que vai de zero até um. O canto superior esquerdo da imagem é o ponto (0, 0), enquanto o inferior direito é o (1, 1). O *Touchlib* limiariza e segmenta a imagem obtida pela *webcam* e a divide em 20 imagens menores. Quando um toque é detectado, sua posição é calculada através de uma interpolação linear apenas na respectiva fatia da imagem. Isso permite que a câmera não necessite estar perpendicular à superfície de projeção, pois a distorção provocada é compensada pela interpolação. A Figura 3 exemplifica este conceito: na sua parte inferior tem-se o detalhe de uma célula da grade em que a superfície de toque é dividida.

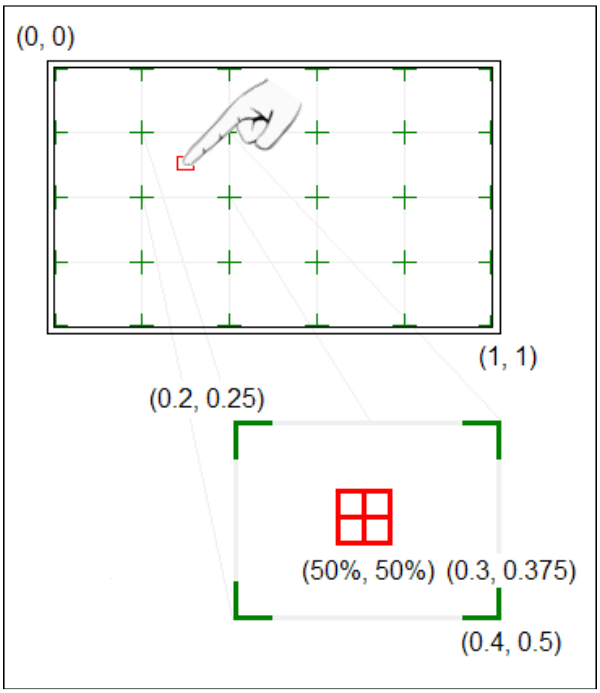

Figura 3: Exemplo de interpolação feita em cada célula no cálculo da posição do toque

5. O projeto IRTaktiks

O jogo *IRTaktiks* é um *RPG* tático, em que o jogador controla diversos personagens com características diferentes, cujo objetivo é derrotar o inimigo através de ataques, magias e itens, utilizando táticas, como por exemplo se beneficiar de uma determinada posição no campo de batalha para obter vantagens sobre o inimigo.

O processamento de eventos de entrada no jogo está exemplificado na Figura 4. As interações do usuário sobre a mesa, por exemplo o toque de um ou mais dedos sobre sua superfície, são reconhecidos pelo *Touchlib* através da análise das imagens enviadas por uma *webcam* posicionada sob a mesa e que enquadra a superfície de interação. O *Touchlib* processa as informações e envia uma mensagem *TUIO* para cada dedo sobre a mesa, contendo as informações como posição, ângulo de movimentação, velocidades calculadas entre outras.

O jogo utiliza a biblioteca *OSCpack* e implementa um cliente *OSC* que recebe as mensagens *TUIO/OSC* geradas pela *Touchlib* e as decodifica para obter informações a respeito dos toques efetuados pelo usuário na mesa. Há um módulo do *IRTaktiks* chamado *Input* que se encarrega de disparar um evento interno ao jogo para cada interação executada pelo usuário na mesa. Estes eventos se propagam para os diversos componentes do jogo que se atualizam conforme necessário. Finalmente, uma imagem do jogo é projetada com o auxílio de um projetor sob a superfície da mesa. O efeito percebido pelo jogador é o de manipular diretamente os objetos do jogo.

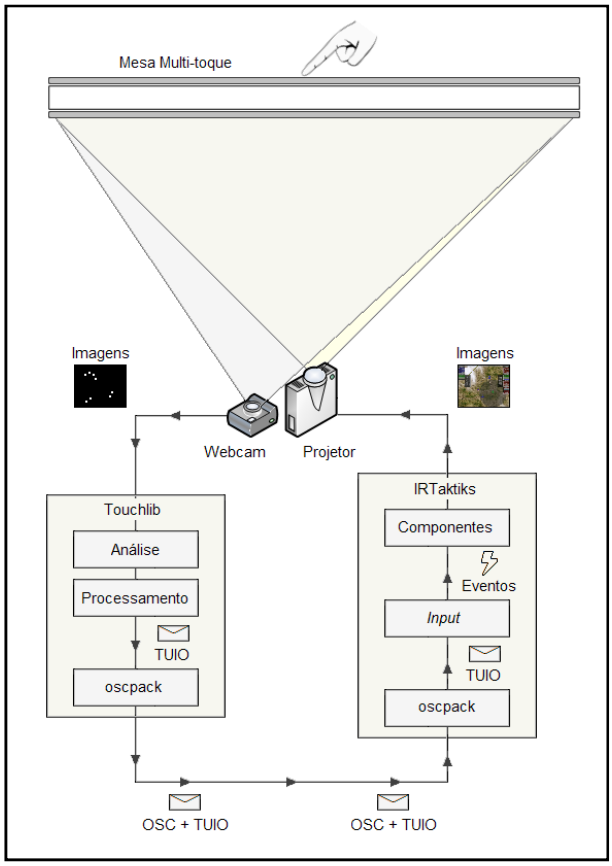

Figura 4: Arquitetura de deteção e processamento de eventos do sistema

5.1 Concepção

O *IRTaktiks* deve ser jogado por dois jogadores, e cada um terá várias unidades de combate. Cada unidade possui diversos atributos que quando configurados a tornam única e diferente das demais. Além de atributos, as unidades possuem classes que lhes dão características, vantagens, desvantagens e ações diferentes ampliando as possibilidades de estratégia de cada um dos usuários. O objetivo é derrotar todas as unidades do jogador adversário, utilizando as características de cada unidade de combate e suas respectivas ações em conjunto com o cenário onde a batalha acontece.

5.2 Mesa multitoque

A mesa multitoque utilizada no *IRTaktiks* é formada por uma superfície de acrílico transparente de aproximadamente 1,2m x 1,6m, acoplada a um suporte de madeira sobre rodas, que facilita seu deslocamento. O acrílico é encaixado numa esquadria de alumínio que possui 47 entradas nas laterais que funcionam como soquetes para *LEDs* infravermelhos, de modo que a luz por eles emitida percorra o interior do acrílico capturada de acordo com o princípio da reflexão total interna da luz. Os *LEDs* infravermelhos utilizados são de alto brilho com corrente máxima suportada de 100mA e tensão de barreira de potencial de 1.2V, subdivididos em 10 trechos de circuito elétrico que estão dispostos ao longo do perímetro do acrílico.

Figura 5: Mesa multitoque utilizada no projeto

Para obter as imagens dos toques foi utilizada uma webcam *Microsoft LifeCam VX-6000*. A escolha desta webcam se deveu a possuir ângulo de visão com 71°, sensor *CCD* (*Charge Coupled Device*) com resolução de 800 por 600 *pixels* e ser capaz de suportar uma taxa de atualização de 30fps (quadros por segundo). A *webcam* veio de fábrica com um filtro que inibe a passagem da luz infravermelha posicionado entre a lente e o *CCD* que teve de ser removido.

A fim de minimizar as influências da luz do projetor e da iluminação ambiente da sala na detecção dos eventos nas imagens adquiridas pela *webcam*, foi adicionado à câmera um filtro que bloqueia a maior parte da luz visível mas permite a passagem de luz infravermelha. Utilizou-se como filtro bloqueador de luz visível um pedaço de filme fotográfico queimado após ter sido revelado, pode-se ver a diferença representada por este filtro na Figura 6.

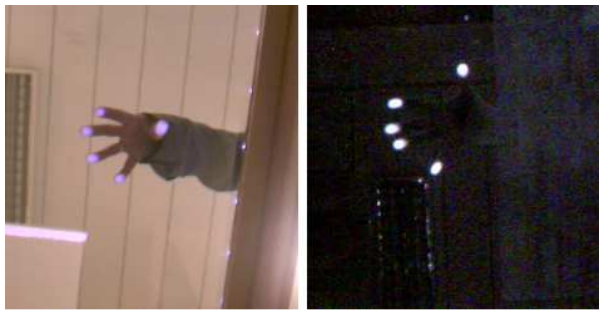

Figura 6: Toque sem e com o filtro inibidor da luz visível

A *webcam* fica posicionada sob a mesa com a superfície de acrílico contida em seu campo de visão, de modo a obter as imagens dos toques realizados pelos usuários. Devido à câmera utilizada ser dotada de um ângulo de visão maior que o de webcams convencionais, pôde ser colocada a uma distância menor em relação ao acrílico e mesmo assim cobrir uma área da mesa maior ou equivalente.

A projeção é feita com um projetor de resolução nativa de 800 por 600 pixels e um espelho simples, como pode ser visto na Figura 7. A imagem do projetor é direcionada pelo espelho para a superfície inferior do acrílico. É necessário posicionar um anteparo de material difusor sob a superfície do acrílico para que o usuário veja a projeção.

Figura 7: Sistema de projeção situado sob a mesa

O material ideal para este tipo de mesa é um polímero para projeções, fabricado pela empresa *Rosco*. Como este material não é encontrado facilmente no Brasil, sua utilização foi descartada. Para substituir o material difusor, foram realizados testes utilizando papel vegetal e sacolas plásticas brancas de polietileno.

Testes realizados indicaram que os sacos plásticos de polietileno permitiram maior nitidez na imagem

capturada pela *webcam* que é usada na detecção dos toques quando comparados com o papel vegetal. Este material não foi encontrado no tamanho necessário para cobrir uma área razoável da mesa sem que fossem necessárias emendas e foi também descartado. Utilizamos papel vegetal como anteparo na maior parte das execuções do protótipo e no vídeo disponibilizado online que mostra o trabalho em execução.

5.3 Arquitetura do software

A utilização do *Touchlib* para reconhecimento de toques e sua compatibilidade com *TUIO/OSC* não representa nenhuma restrição ou condição de contorno em relação a escolhas relacionadas à bibliotecas gráficas. A parte visual do jogo poderia ser desenvolvida em praticamente qualquer plataforma de computação gráfica ou *API* 3D, pois existem diversas implementações do protocolo *OSC*, em diversas plataformas e linguagens de programação. Dessa forma, a escolha foi baseada principalmente em qual ambiente a produtividade seria maior e qual proveria mais recursos, como controle de versões, gerenciadores de conteúdo e linguagens com suporte a programação orientada a objetos. Dentre frameworks existentes, optou-se pelo *Microsoft® XNA 2.0*, devido à experiência anterior dos integrantes do projeto com o ambiente de desenvolvimento da *Microsoft* com *C#*, grande variedade de recursos disponíveis, ampla documentação, desempenho do código gerado e possibilidade de utilizar o ambiente integrado de desenvolvimento voltado à produtividade, *Microsoft Visual Studio*.

Após a definição da plataforma em que o jogo seria desenvolvido, iniciou-se a construção de um módulo chamado *Listener*, responsável pela comunicação entre o jogo e o software que controla a detecção dos toques sobre a mesa (*Touchlib*). Dessa forma, futuros problemas de integração seriam eliminados, uma vez que a construção do jogo levaria este módulo de comunicação em consideração, sem alterá-lo. Foi decidido que este módulo utilizaria eventos internos para representar as interações dos usuários com a mesa. Desta forma o projeto do jogo foi simplificado e modularizado. O serviço que lê as informações contidas nas mensagens *OSC/TUIO* e dispara os eventos é executado em uma *thread* à parte; aumentando o desempenho do módulo de comunicação.

Este módulo foi construído utilizando as bibliotecas do *OSCpack*, que é responsável por obter as mensagens *TUIO* enviadas pela mesa, decodificá-las e transformá-las em entradas para o jogo através da comunicação com o módulo Input. Baseia-se em uma arquitetura cliente-servidor, exercendo a função de cliente.

As mensagens *TUIO* possuem informações sobre cada um dos toques e objetos que estão sobre a mesa. Cada cursor ou objeto possui um identificador, que é de base para o reconhecimento de ações mais complexas como, por exemplo, funcionalidades de arrastar e soltar.

Além de identificadores, cada cursor e objeto posicionado sobre a mesa possui três tipos de mensagens diferentes: *Down*, *Update* e *Up*.

As mensagens *Down* são enviadas quando o objeto ou o cursor são criados, ou seja, quando o objeto é colocado sobre a mesa ou quando o dedo do jogador encosta sobre sua superfície. As mensagens *Update* são enviadas para informar que o cursor ou o objeto estão ativos, ou seja, servem para informar que o objeto continua sobre a mesa, parado ou em movimento. Já as mensagens do tipo *Up* são enviadas quando o objeto ou o cursor não estão mais disponíveis e foram removidos da superfície da mesa. Com estes três tipos de mensagens é possível rastrear qualquer tipo de movimento sobre a mesa, seja ele usando objetos, toques, ou até mesmo uma combinação de ambos.

A arquitetura do jogo foi projetada de modo a deixar o *IRTaktiks* o mais leve e rápido possível, além de possibilitar a agregação de novas funcionalidades rapidamente e de maneira robusta. Utilizando reuso de módulos, processamento da placa de vídeo em conjunto com o do computador, *cache* de texturas e imagens e atualizações lógicas dos componentes somente quando necessário foi possível obter uma boa velocidade de execução sem comprometer a cobertura dos requisitos propostos ou a facilidade de manutenção do código fonte.

Dividiu-se a arquitetura em diversos módulos a fim de facilitar a implementação e extensão de funcionalidades, uma vez que com padrões definidos, a agregação de novas funcionalidades é bastante fácil e ágil. O diagrama da Figura 8 ilustra os módulos presentes no *IRTaktiks*.

O módulo *Listener* é responsável pela decodificação das mensagens *TUIO/OSC*, enviando as informações para o módulo *Input*, que dispara os eventos de adição, movimentação e remoção de dedos sobre a mesa. O módulo *Resource* efetua o carregamento dos recursos gráficos que serão desenhados pelo módulo *Drawable*, como texturas, imagens, partículas, efeitos e fontes. O módulo *Game* é a representação dos objetos do jogo, por exemplo, os jogadores e suas unidades. Cada unidade possui características que são descritas pelo módulo *Logic*, ações que são implementadas no módulo *Action* e menus que são construídos pelo módulo *Menu*. A interação entre as ações e os menus é realizada pelo módulo Interaction. O módulo *Screen* implementa as várias telas do jogo e suas transições, enquanto o módulo *Debug* é utilizando para auxiliar o desenvolvimento e testes de novas funcionalidades.

A arquitetura interna no *XNA* é centralizada na classe *Game*, que provê métodos para atualização e desenho de objetos, além de possuir uma lista de *GameComponents* e *Services*, que são atualizados e

desenhados automaticamente pela classe Game. Internamente, o *XNA* cria uma *thread* para cada componente e serviço e não há uma ordem de execução pré-especificada. A vantagem desta arquitetura é a velocidade na execução, uma vez que várias *threads* executando paralelamente se beneficiam dos processadores *multi-core*, bastante comuns hoje em dia.

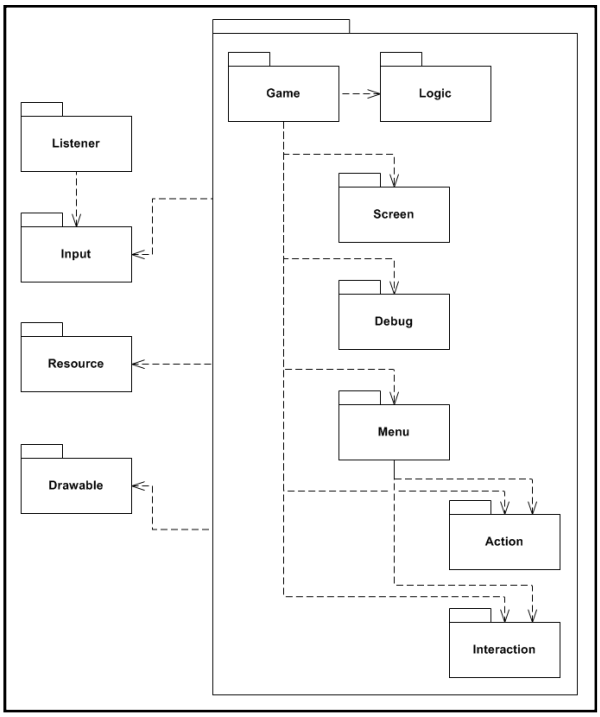

Figura 8: Arquitetura da versão final

6. Resultados

6.1 Funcionamento do jogo

O *IRTaktiks* é jogado por dois jogadores competindo entre si, que terão a disposição unidades de combates customizáveis. O controle destas unidades é feita através de *gestures*: para mover uma unidade basta tocar a unidade e arrastar o dedo pela superfície da mesa. Através do toque também são feitas as seleções de menu. Todas as ações podem ser executadas paralelamente por ambos os jogadores, com toques suaves, permitindo mais naturalidade ao jogar.

Não foi tomada nenhuma precaução específica relacionada a controle de acesso às unidades, ou seja, qualquer um que toque em determinada unidade poderá comandá-la. Considera-se que seja aceitável este funcionamento do jogo, pois este problema também está presente em jogos de tabuleiro convencionais como o xadrez, em que os jogadores ativamente obedecem às regras limitando-se a procurar controlar apenas suas peças.

Conforme se observa na Figura 9, a área de jogo é divida entre as unidades (1), os menus (2) e o mapa (3).

As unidades podem se movimentar pelo mapa e executar ações, controladas através dos menus laterais. As informações de cada unidade e dos jogadores também são exibidos neste menu. O posicionamento das unidades pelo mapa é um dos principais fatores de estratégia do jogo. A posição, de acordo com a altura do terreno, em combinação com as características das unidades, afeta o campo de visão e o alcance das habilidades.

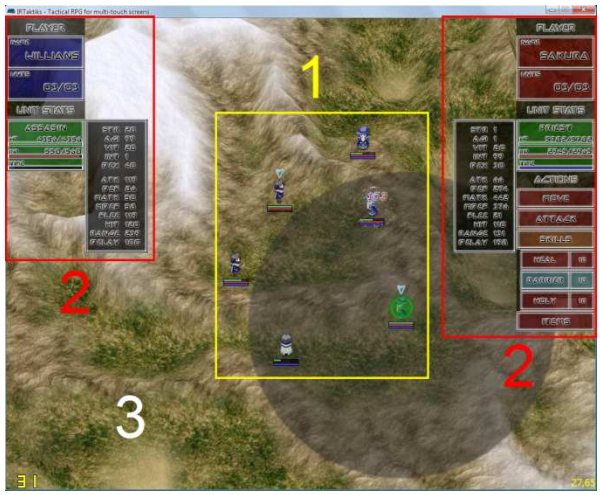

Figura 9: Elementos do jogo.

Outro fator que permite a estratégia é a personalização de cada uma das características das unidades que o jogador possui. Uma unidade possui características básicas que influenciam em desde seu ataque, velocidade de ação, defesa, magia e destreza até porcentagem de desvio e ataque, pontos de vida e mana, como podem ser visto na Figura 10. Estes atributos de cada unidade são configurados pelos dois jogadores simultaneamente numa tela pré-jogo.

Figura 10: Características de uma unidade

Além de características, as unidades também possuem tipos, que determinam as habilidades que a unidade vai possuir e quais características serão mais influentes. Sua combinação com as características básicas permite a construção de uma unidade voltada para ataque, defesa ou suporte de personagens, aumentando as possibilidades de jogo e uso de táticas.

Cada tipo de unidade possui ações específicas que podem ser disparadas sob o comando de seu jogador. Estas ações foram escolhidas de acordo com as características mais influentes, de modo a efetuar um balanceamento entre as unidades e não deixar um tipo mais forte que outro em todas as circunstâncias. Um exemplo do menu de ações com as habilidades de uma unidade pode ser visto na Figura 11.

Figura 11: Menu de ações de uma unidade

Todas as ações de um mesmo tipo seguem um mesmo padrão de utilização por parte do jogador. Há três tipos de ações: movimentação, uso e bônus. As ações de movimentação requerem que a unidade se movimente pelo mapa. Já as de uso necessitam que o jogador escolha uma unidade no mapa para ser o alvo da habilidade que será executada. Finalmente, as ações de bônus têm como alvo a própria unidade que a invocou, não necessitando de uma utilização especial por parte do jogador.

Quando uma ação de movimentação é escolhida, uma área circular é desenhada em volta da unidade, determinando os limites de movimento da mesma. De posse dessa informação, o jogador pode mover a unidade para qualquer posição dentro da área delimitada, como pode ser visto na Figura 12.

Quando uma ação de uso é selecionada, uma área é desenhada em volta do personagem determinando o limite de uso da habilidade escolhida e uma mira é criada. A escolha do alvo é feita arrastando essa mira sobre a unidade alvo. Se ela estiver sobre um aliado, sua cor será verde, enquanto sobre um inimigo ela ficará vermelha, como na Figura 13.

Figura 12: Unidade se movimentando dentro de uma área

Quando a mira é solta, a ação escolhida no menu é executada. Todas as ações são graficamente representadas com animações e as informações sobre modificações nos status das unidades são exibidos, como na Figura 14. Quando uma unidade recebe um bônus, a informação é exibida em verde, enquanto a informação de danos é mostrada em vermelho. Quando os pontos de vida de uma unidade chegarem a zero ela desmaia e se torna inativa. O jogo encerra quando um jogador conseguir deixar todas as unidades de seu adversário desmaiadas.

Figura 13: Mira sobre uma unidade inimiga

Quando uma ação é executada, independentemente de seu tipo, ela é regida por um fluxo, que determina como a ação deve ser executada. Isso se dá ao fato da ação poder ser executada sobre nenhuma unidade, ou ainda aplicar efeitos não instantâneos, ou seja, que duram um determinado periodo de tempo. O fluxo de execução de uma ação é exibido na figura 15.

Figura 14 - Exemplo de exibição de informações

6.1 Funcionamento do jogo

O *IRTaktiks* é executado em dois computadores: um deles tem a webcam e executa a *Touchlib*, que usa os protocolos *TUIO/OSC* para enviar os dados da interação ao módulo principal que executa em uma outra estação. Nos testes realizados utilizou-se um *Celeron* com 2.0 GHz para executar a *Touchlib* e uma estação com um *Intel Core 2 Duo* e placa gráfica *NVidea GeForce 8800* para execução da parte visual, que precisa ser compatível com *DirectX Pixel Shaders* 3.0 para poder gerar a malha 3D do terreno e executar um shader que define sua aparência com base em um mapa de alturas. Estes *shaders* não são um requisito para suportar a maior parte da funcionalidade do *IRTaktiks*, de modo que pode ser reescrito para executar em uma configuração mais modesta.

Cerca de 30 pessoas jogaram o *IRTaktiks* no dia 03/10 num evento no *Centro Universitário Senac*, e observou-se que a familiarização de novos usuários com os conceitos do jogo é bastante rápida. Depois de receber uma explicação sucinta sobre o que é o jogo e quais os objetivos e verem outros jogadores em ação, a maior parte das pessoas já era capaz de jogar rudimentarmente mesmo sem muita familiaridade com os diversos tipos de unidades, atributos e táticas permitidas pelo jogo.

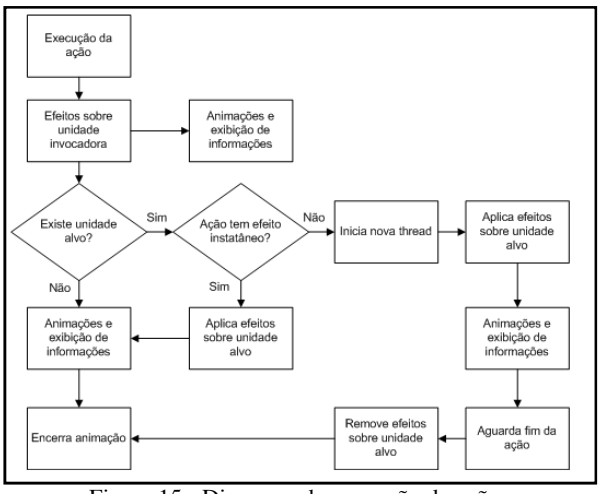

Figura 15 - Diagrama de execução de ações

7. Conclusões e Trabalhos Futuros

Durante a realização deste trabalho, percebeu-se a importância de uma adequada gerência nas interações dos diversos usuários da interface multi-toque. Ignorar este quesito no desenvolvimento de uma aplicação para este fim pode transformar a interação, que deveria ser fácil e natural, em algo difícil e cansativo. Na versão aqui apresentada supunha-se que apenas o jogador que é o legítimo dono de uma unidade tentaria comandá-la, mas às vezes por uma confusão sem má intenção por parte dos jogadores esta regra falhava em ser seguida.

Os principais objetivos traçados durante a concepção do jogo foram atendidos. Dois jogadores controlam suas respectivas unidades em batalhas, em que as características únicas de cada unidade, aliadas à tática são os fatores decisivos para derrotar o adversário. A interface natural e o bom *feedback* do jogo às ações do usuário também faz com que seja divertido mesmo para jogadores iniciantes.

Agradecimentos

Os autores gostariam de agradecer a João J. Maranhão Jr. pela construção da estrutura da mesa e pela primeira versão de sua parte elétrica. Agradecemos também ao Centro Universitário Senac por viabilizar com sua infra-estrutura o desenvolvimento do projeto.

Referências

HAN, J. Y. Low-Cost Multi-Touch Sensing through Frustrated Total Internal Reflection. *Proceedings of the 18th Annual ACM Symposium on User Interface Software and Technology*, 2005.

BUXTON, Bill. *Multi-Touch Systems that I Have Known and Loved*. Disponível em: http://www.billbuxton.com/multitouchOverview.html (Acessado em 06 de outubro de 2008).

MULTIGESTURE.NET. *A Multi-Touch and Multi-Gesture research blog*. Disponível em: http://www.multigesture.net/ (Acessado em 06 de outubro de 2008).

NUIGROUP. *Touchlib: A multitouch Development Kit*. Disponível em: http://nuigroup.com/touchlib (Acessado em 06 de outubro de 2008).

CNMAT, UC Berkeley. *Open Sound Control*. Disponível em: http://opensoundcontrol.org/ (Acessado em 06 de outubro de 2008).

BENCINA, Ross. *OSCPack*. Disponível em: http://www.audiomulch.com/~rossb/code/oscpack/ (Acessado em 06 de outubro de 2008).

TUIO. *TUIO: A Protocol for Tangible User Interfaces*. Disponível em: http://tuio.lfsaw.de (Acessado em 06 de outubro de 2008).

KALTENBRUNNER, MARTIN, ET AL. *TUIO: A Protocol for Table-Top Tangible User Interfaces*. Proceedings of the 6th International Workshop on Gesture in Human-Computer Interaction and Simulation (GW 2005). Vannes, France, 2005.

KALTENBRUNNER, MARTIN, BENCINA. *reacTIVision: A Computer-Vision Framework for Table-Based Tangible Interaction*. Proceedings of the First International Conference on Tangible and Embedded Interaction (TEI 2007). Baton Rouge, Louisiana, 2007.

KALTENBRUNNER, M. & JORDÀ, S. & GEIGER, G. & ALONSO, M. *The reacTable: A Collaborative Musical Instrument*. Proceedings of the Workshop on "Tangible Interaction in Collaborative Environments" (TICE), at the 15th International IEEE Workshops on Enabling Technologies (WETICE 2006). Manchester, U.K