IRTaktiks: Jogo de Estratégia para mesa Multitoque

Paper 49249

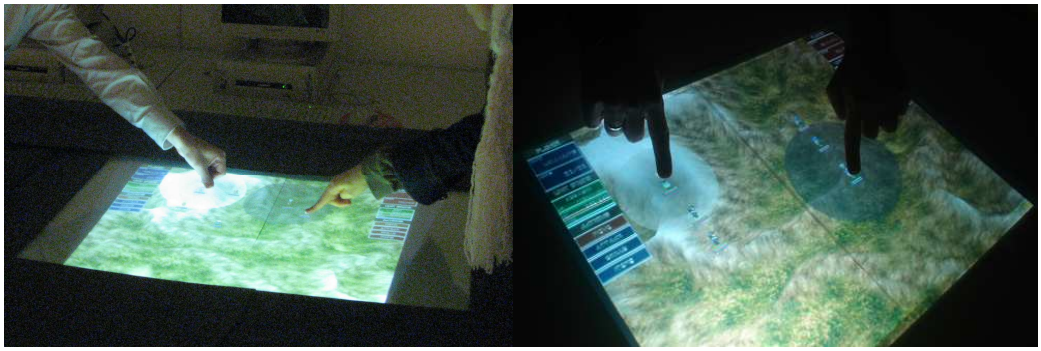

Figura 1: IR Taktiks jogado simultaneamente por dois jogadores, que manipulam suas unidades diretamente com as mãos

Abstract

This work describes the design and implementation of a real-time multiplayer strategy game called IRTaktiks that is playable on a multi-touch table. Is comprises all stages of work ranging from details of the physical interaction device, game design and implementation and presentation of its results.

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

Authors' contact:
`Omitted for blind review`

Sample Video:
www.tinyurl.com/irtaktiks/

1. Introdução

Em tempos recentes tem havido interesse crescente em dispositivos de interação multitoque, caracterizados por um ou mais usuários podem interagir através dos dedos diretamente com uma interface gráfica, dispensando o uso de dispositivos de entrada mais convencionais, como teclados ou canetas especiais. Inicialmente protótipos de telas e mesas de interação multitoque desenvolvidos por pesquisadores de interação e entusiastas foram demonstrados, e subseqüentemente percebeu-se um surgimento de produtos computacionais que embutem este conceito, por exemplo, o telefone iPhone ou o 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 meio para colaboração quer na forma 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 nos desktops normais por questões ergonômicas (apenas um mouse e teclado, mas que muitas vezes está presente em jogos de mesa e tabuleiro.

Quer na forma de mesas ou outros tipos de dispositivos, são razoáveis 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*, de 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 alguns aspectos relacionados à construção do dispositivo de interação, detecção dos multitoques 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.

A organização deste artigo é descrita a seguir. Na próxima seção alguns trabalhos 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

Esta tecnologia se popularizou com a ajuda do *YouTube*, em 2006; com vídeos do evento *Technology Entertainment Design Conference*, em *Monterey* na Califórnia. Nele, o pesquisador do instituto de ciências matemáticas *Courant* , *Jeffenson Y. Han*; demonstrou

seu trabalho de pesquisa de interação multitoque utilizando uma superfície com display gráfico interativa, permitindo a interação de múltiplos usuários; apresentando implementações de diversas técnicas e aplicações, sendo alguma delas jogos multitoque. Os protótipos de *J. Han* [Han 2005] despertaram o interesse de diversas vertentes de pesquisa sobre essa nova alternativa de interação, populando a Internet com diversos tutoriais e *weblogs* [Buxton 2008] que trocam experiências entre estes pesquisadores e fomentam o desenvolvimento de protótipos.

O multitoque teve seu início em 1982, 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: aponte e arraste com um ou mais dedos (impressas sobre uma superfície macia), eliminando totalmente a necessidade de um dispositivo apontador, como o mouse.

O primeiro dispositivo multitoque com display visual integrado comercializado foi o *Lemur Input Device*, um controlador multimídia profissional da companhia francesa *JazzMutant* lançado em 2005. 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 *ReacTable* [Kaltenbrunner et al. 2006] , é um instrumento musical colaborativo, que permite o reconhecimento de objetos postos sobre sua superfície e tem a possibilidade de detectar interação multiusuário e sintetizar sons através de uma cadeia de fontes, filtros e osciladores, 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. Assim, 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 dos objetos sobre a mesa. Foi utilizado em shows e apresentações da cantora islandesa *Björk*, no *Coachella* Festival, em 2007 na Califórnia.

3. FTIR e Multitoque

Atualmente, existem diversas técnicas para a detecção de múltiplos toques em superfícies, desde a análise de uma imagem pura, 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 complexa utilização de circuitos capacitores que armazenam a energia elétrica emitida pelo corpo humano quando o toque ocorre.

As mais técnicas mais simples e baratas utilizam 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 com luz infravermelha.

A mesa multitoque utilizada neste trabalho (que pode ser vista na Figura 5) foi construída baseando-se nas técnicas de 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, todas as partículas do feixe de luz são redirecionadas ao meio de propagação de origem, ao contrário da parcial, onde algumas partículas atravessam a interface entre os meios de propagação, ocorrendo um desvio no ângulo de incidência do feixe de luz emitido, chamado de refração.

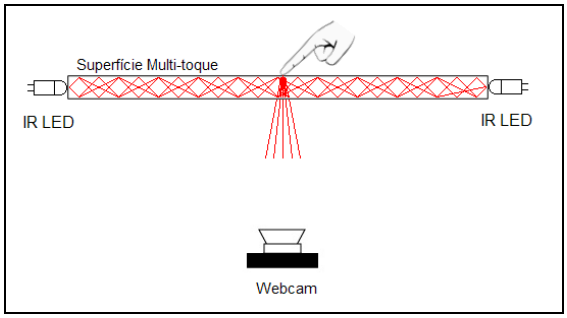

Figura 2 - Reflexão total interna frustrada da luz

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.

Transpondo a teoria para a aplicação em interfaces multitoque, 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 brancos na imagem capturada e a posição dos toques é facilmente detectada. Pode-se ver este fenômeno em efeito na Figura 6.

4. Infra-Estrutura e Ferramentas

Para o reconhecimento de toques sobre a superfície da mesa multi-toque, alguns softwares publicamente

disponíveis e desenvolvidos por terceiros foram utilizados. Existe uma grande quantidade de superfícies multi-toque 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*. Havia também a necessidade de fazer o *reacTIVision*, software responsável pela identificação de toques e fiduciais, enviar informações para o aplicativo que gerava os sons e efeitos. Para efetuar esta comunicação de mensagens do protocolo *TUIO*, a equipe da *reacTable* decidiu utilizar o protocolo *OSC* (*Open Sound Control*), que permitia ser meio de transporte para outros protocolos, desta forma a informação *TUIO* é encapsulada em pacotes *OSC*.

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.

O software de reconhecimento do toques *Touchlib*, foi o escolhido para o controle da mesa multi-toque devido à sua estabilidade, número de funcionalidades e uso dos padrões propostos. Isso permite que o jogo funcione corretamente em qualquer superfície multitoque que siga os padrões propostos, aumentando sua interoperabilidade com outros projetos. A seguir serão fornecidos mais alguns detalhes a respeito de *TUIO*, *OSC* e *Touchlib*.

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

OSCpack 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*, *AudioMulch*, entre outros; principalmente pela capacidade de prover a comunicação entre as plataformas *Windows*, *Linux* e *Mac*.

TUIO é um protocolo de comunicação desenvolvido com a finalidade de atender os requisitos de comunicação entre interfaces tangíveis. Define propriedades comuns baseadas no controle de objetos, toques e gestos. Este protocolo foi criado pela equipe de desenvolvimento do projeto *ReacTable* e implementado sobre o protocolo de comunicação *OSC*. Hoje, possui diversas implementações nas linguagens *Java*, *C*, *C++*, *Flash*, entre outras.

As mensagens são divididas em profiles, baseados na interação com a interface tangível. Atualmente, possui o *TUIO* possui profiles para interfaces 2D, 3D e customizadas. Cada um, 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:. sessão, identificador da interação, posição no espaço 2D ou 3D, ângulo, vetor de movimento, vetor de rotação, aceleração de movimento, aceleração de rotação.

O *Touchlib* é uma biblioteca que permite a detecção de toques em superfícies multi-toque que utilizam o princípio da Reflexão Total Interna Frustrada da Luz, Iluminação Traseira ou Iluminação Frontal. Esta biblioteca foi desenvolvida pela *Natural User Interface Group* em parceria com a *White Noise Audio*, e é bastante utilizada em aplicações multi-toque devido ao grande número de funcionalidades.

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 quando um dedo toca, percorre ou é retirado da superfície multitoque. Esta biblioteca permite a integração com demais aplicativos através do protocolo *TUIO*, sob o protocolo *OSC*, utilizando a biblioteca *oscpack*.

O *Touchlib* aplica filtros nas imagens recebidas, 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* e *Linux*.

O *Touchlib* trabalha em 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 de imagem. Isso permite que a câmera não necessite estar perpendicular à superfície de projeção, pois a distorção provocada é compensada via software.

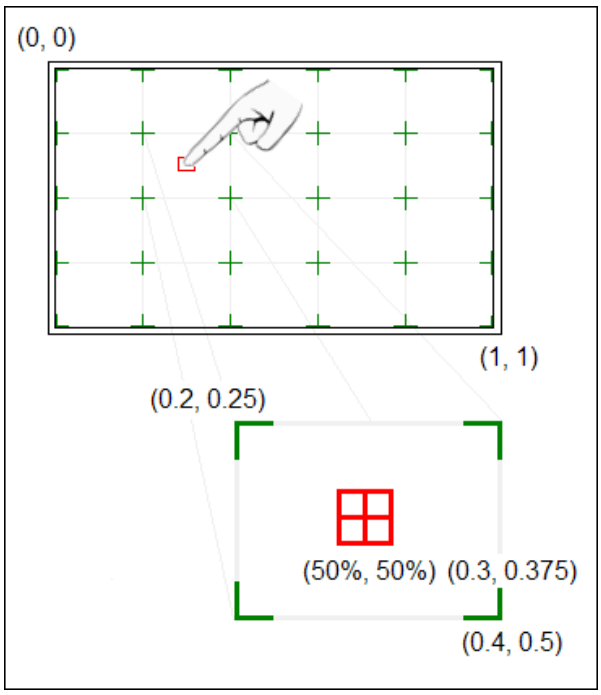

Figura 3 - Exemplo de interpolação no cálculo da posição do toque

5. O projeto IRTaktiks

O jogo desenvolvido é um *RPG* tático, em que o jogador controla vários 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.

As ações que os usuários executarem sobre a mesa, como o toque de um ou mais dedos sobre sua superfície, será reconhecida pelo *Touchlib* através da análise das imagens enviadas por uma *webcam*. 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.
Estas mensagens *TUIO* são empacotadas dentro de envelopes *OSC* e enviadas através da biblioteca *oscpack*, para a aplicação cliente, no caso o jogo. O jogo decodifica o envelope *OSC*, utilizando também a biblioteca *oscpack*; e obtém a mensagem *TUIO* original, juntamente com as informações sobre os toques.

Para cada ação efetuada na mesa, um evento é disparado pelo módulo *Input*. Estes eventos sensibilizam os diversos componentes do jogo que se atualizam. Finalmente, o jogo é projetado com o auxílio de um projetor sob a superfície da mesa. Com isso, o usuário possui a impressão de estar manipulando diretamente os objetos do jogo.

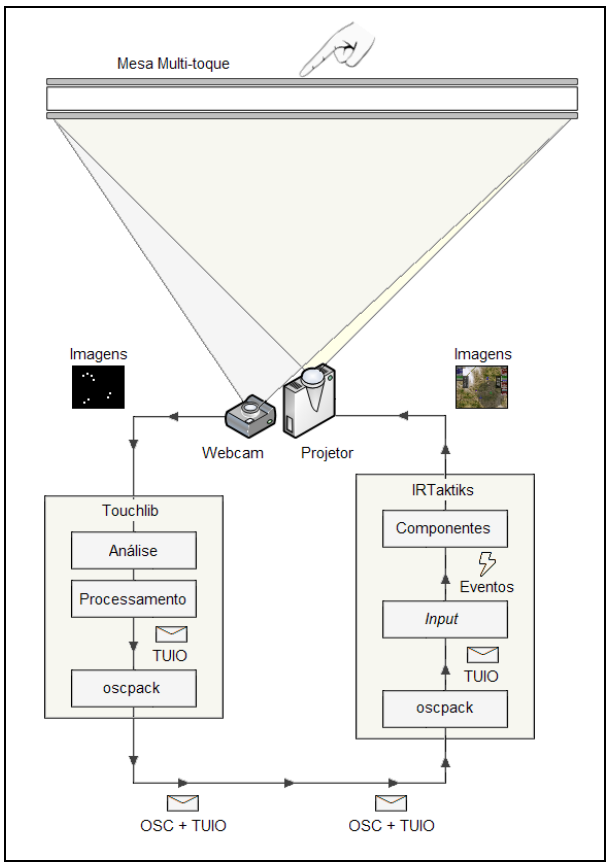

Figura 4 - Arquitetura do sistema

5.1 Concepção

O jogo deve ser jogado por dois jogadores, sendo que cada jogador terá várias unidades de combate. Cada uma possui diversos atributos que quando configurados tornam-na única e diferente das demais em vários aspectos. 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 times. 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 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. Possui 47 entradas para LEDs infravermelhos, de modo que a luz percorra o interior do acrílico capturada de acordo com o princípio da reflexão total interna frustrada da luz. Estas entradas são dispostas pelos quatro lados da mesa, intensificando a propagação da iluminação dos LEDs, em direção ao centro do acrílico.

Figura 5 - Mesa multitoque utilizada no projeto

Para facilitar a manutenção, evitando dificuldades de reparos após eventuais novas depredações, decidimos tornar todas as ligações completamente modulares e de fácil substituição, pois não seria utilizada nenhuma solda ou cola na fixação dos componentes

A mesa conta com 47 *LEDs* infravermelhos de alto brilho, com corrente elétrica de trabalho de 100mA e tensão de barreira de potencial de 1,2V, subdivididos em 10 ramos. Cada ramo possui dois resistores: um de 56Ω e outro de 5,6Ω ligados em série, limitando a corrente de 5 *LEDs* ligados em série. O último ramo por possuir apenas 2 *LEDs*, conta com dois resistores de 56Ω ligados em série.

Para obter as imagens dos toques foi utilizada uma webcam *Microsoft LifeCam VX-6000*. A escolha desta webcam se deu ao fato de possuir ângulo de visão com 71º, maior que o da maioria das webcams disponíveis no mercado, sensor *CCD* (*charge coupled device*) com resolução de 800px por 600px e taxa de atualização de 30fps (quadros por segundo).

Esta *webcam* veio de fábrica com um filtro que inibe a passagem da luz infravermelha. Para a correta utilização neste projeto, este filtro teve que ser removido para que as imagens dos toques, que somente são visíveis ao espectro de luz infravermelha; pudessem ser passadas ao *Touchlib*.

Para que pudéssemos ter uma imagem dos toques sem influências da iluminação externa, foi adicionado à câmera um filtro que bloqueia a maior parte da luz visível, mas permite a passagem de luz infravermelha. O filtro utilizado foi um filme fotográfico.

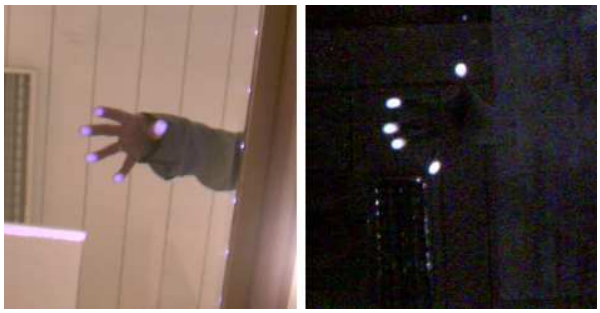

Figura 6 - Toque sem e com o filtro inibidor da luz visível

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, pode ser colocada a uma distância maior 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 800px por 600px e um espelho. Como pode ser visto na Figura 45, a imagem é projetada num espelho que redireciona a imagem 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

O material ideal para este tipo de mesa é um polímero para projeções, fabricado pela *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 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 e foi também descartado. Devido a contingências na aquisição de filmes de polietileno, o papel vegetal foi adotado como anteparo de projeção nos resultados descritos posteriormente, apesar de seu desempenho inferior.

5.3 Jogo

Por utilizamos o software *Touchlib* para o reconhecimento de toques, e pelo fato deste utilizar uma arquitetura de comunicação baseada no protocolo *TUIO* juntamente com o *OSC*, através da biblioteca, oscpack; o jogo poderia ser desenvolvido 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, escolhemos o *Microsoft® XNA 2.0*, devido à enorme variedade de recursos disponíveis, documentação, desempenho e ganho de produtividade, uma vez que a *IDE* de desenvolvimento seria o *Microsoft Visual Studio* e linguagem de programação adotada *C#*.

Após a escolha do ambiente de desenvolvimento, iniciou-se a construção de um módulo de comunicação entre o jogo e o software que controla a detecção dos toques sobre a mesa (*Touchlib*) foi projetado e desenvolvido. 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 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*, 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, servido de base para o reconhecimento de ações mais complexas como funcionalidades *drag-and-drop*, ou simplesmente arrastar e soltar. Além de identificadores, cada cursor e objeto possuem 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 encosta sua superfície. As mensagens *Update* são enviadas para informar que o cursor ou o objeto estão ativos, em outras palavras, servem para informar que o objeto continua sobre a mesa, parado ou em movimento, ou ainda para informar que o mesmo dedo encontra-se sobre a mesa, também parado ou em movimento. Já as mensagens to tipo *Up* são enviadas quando o objeto ou o cursor são removidos, ou seja, quando 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.

O desenvolvimento do jogo teve como foco principal sua arquitetura. 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, cachê de texturas e imagens, e atualizações lógicas dos componentes somente quando necessário; obteve-se velocidade de execução e manutenção, sem comprometer a qualidade dos requisitos propostos.

A arquitetura foi dividida em diversos módulos, a fim de facilitar a implementação e extensão de funcionalidades, uma vez que com padrões definidos, a adição de novas funcionalidades é bastante fácil e ágil.

O módulo *Listener* é responsável pela decodificação das mensagens *OSC+TUIO*, 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, como 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.

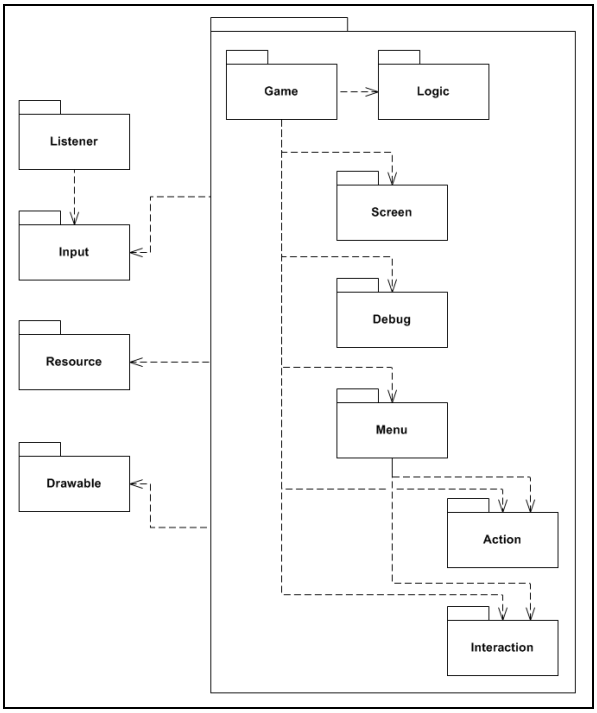

Figura 8 - Arquitetura da versão final

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, não havendo, portanto, uma ordem prevista de execução. 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.

6. Resultados

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.

De acordo com a 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.

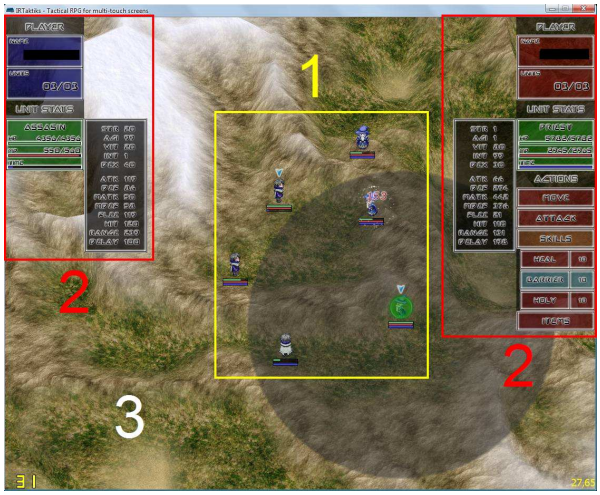

Figura 9: Elementos do jogo.

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.

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 pode ser visto na figura 10.

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 focada para ataque, defesa ou suporte de personagens, aumentando as possibilidades de jogo e uso de táticas.

Como dito anteriormente, cada tipo de unidade possui ações específicas. Estas ações foram escolhidas de acordo com as características mais influentes, de modo a efetuar um balanceamento entre as unidades, não deixando um tipo mais forte que outro. 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

Dentre as ações possíveis, todas utilizam um padrão de utilização por parte do jogador. Todas as ações podem ser divididas em três tipos: 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.

Figura 12 – Unidade se movimentando dentro de uma área

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 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.

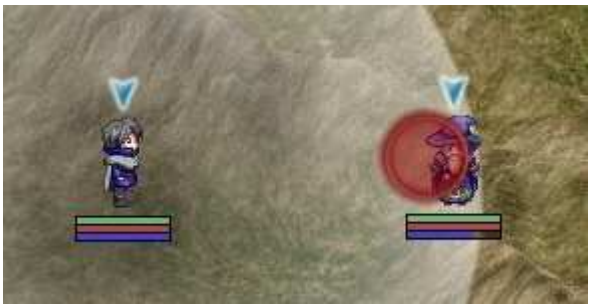

Figura 13 - Mira sobre uma unidade inimiga

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 chegar a zero ela desmaia. O jogo encerra quando um jogador conseguir deixar todas as unidades de seu adversário desmaiadas.

Figura 14 - Exemplo de exibição de informações

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.

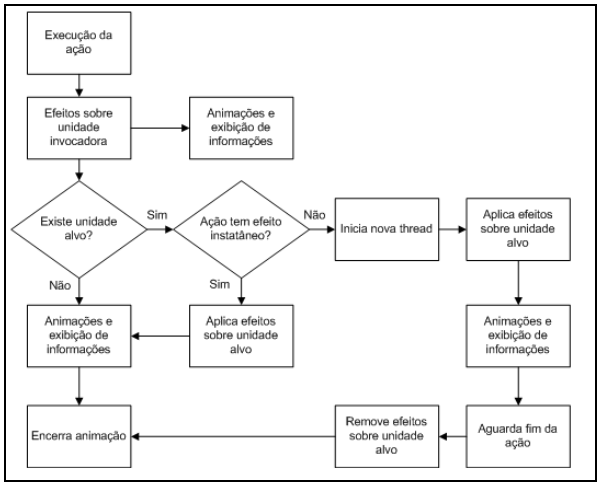

Figura 15 - Diagrama de execução de ações

7. Conclusões e Trabalhos Futuros

Durante a realização deste trabalho, percebeu-se da 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.

As principais características definidas durante a concepção do jogo foram atendidas. Dois jogadores controlam suas respectivas unidades em batalhas, onde as características únicas de cada unidade, aliadas à tática são os fatores decisivos para derrotar o adversário.

A utilização da área total da superfície da mesa para interação com o jogo só não foi possível devido à falta de anteparos difusores que cobrissem sua total extensão. A indisponibilidade do material ideal para esse tipo de superfície nos levou a alternativas que não proporcionaram a máxima qualidade na projeção e detecção de toques, limitando a área de uso de sua superfície.

Referências

HAN, J. Y. Low-Cost Multi-Touch Sensing through Frustrated Total Internal Reflection. **In: 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> Acesso em: 19 de junho de 2008.

MICROSOFT. **MICROSOFT SURFACE.** Disponível em: <http://www.microsoft.com/surface/index.html> Acesso em: 19 junho 2008.

MICROSOFT. **Windows Vista Blog:** Microsoft demonstrates Multi-touch. Disponível em: <http://windowsvistablog.com/blogs/windowsvista/archive/2008/05/27/microsoft-demonstrates-multi-touch.aspx> Acesso em: 19 junho 2008.

WIRED. **ReacTable Tactile Synth Catches Björk's Eye -- and Ear.** Disponível em: <http://www.wired.com/entertainment/music/news/2007/08/bjork_reacTable> Acesso em: 19 de junho de 2008.

MUSIC TECHNOLOGY GROUP, Pompeu Fabra University. **Reactable.** Disponível em: <http://reactable.iua.upf.edu/> Acesso em: 19 de junho de 2008.

FILE. **Interactive Sonic Systems Team**. Disponível em: <http://www.filefestival.org/site_2007/pop_trabalho.asp?id_trabalho=1839&cd_idioma=1&acao=visualizar>. Acesso em: 19 de junho de 2008.

KESTREL, Gwendolyn F.M. **Working Hard at Play.** Disponível em: <http://www.newhorizons.org/strategies/literacy/kestrel.htm> Acesso em: 10 de junho de 2008.

BIMBER, Oliver; RASKAR, Ramesh. **Spatial Augmente Reality**: merging real and virtual worlds. Wellesley, MA: A K Peters, 1997.

KIRNER, Cláudio; TORI, Romero**. Realidade Virtual**: Conceitos e Tendências. São Paulo: Editora Mania de Livro, 2004.

UNIVERSIDADE FEDERAL DO RIO DE JANEIRO. **Augmented Reality**: Realidade Aumentada e Visão Computacional. Disponível em: <http://www.lamce.ufrj.br/grva/realidade_aumentada/> Acesso em: 15 de maio de 2006.

AZUMA, Ronal T. A survey of Augmented Reality. **In: Presence: Teleoperators and Virtual Environments**. v. 6. p 355-385. ago 1997.

MULTIGESTURE.NET. **A Multi-Touch and Multi-Gesture research blog.** Disponível em: <http://www.multigesture.net/> Acesso em: 19 de junho de 2008.

WHITE NOISE AUDIO. **White Noise Audio Software.** Disponível em: <http://www.whitenoiseaudio.com/> Acesso em: 19 de junho de 2008.

NUI GROUP. **NUI Group.** Disponível em: <http://nuigroup.com/> Acesso em: 19 de junho de 2008.

HAN, J. Y.. **NYU Courant Institute of Mathematical Sciences.** Disponível em: <http://www.cs.nyu.edu/~jhan/> Acesso em: 19 de junho de 2008.

CNMAT, UC Berkeley. **Open Sound Control.** Disponível em: <http://opensoundcontrol.org/> Acesso em: 19 de junho de 2008.

BENCINA,Ross. **OSCPack.** Disponível em: <http://www.audiomulch.com/~rossb/code/oscpack/>. Acesso em: 19 de junho de 2008.

TUIO. **TUIO: A Protocol for Tangible User Interfaces**. Disponível em: <http://tuio.lfsaw.de>. Acesso em: 19 de junho de 2008.

KALTENBRUNNER, Martin, et All. **TUIO:** A Protocol for Table-Top Tangible User Interfaces.

KALTENBRUNNER, Martin, BENCINA. **reacTIVision: A Computer-Vision Framework for Table-Based Tangible Interaction**.

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

NUIGROUP. **Touchlib: A multitouch Development Kit**. Disponível em: <http://nuigroup.com/touchlib>. Acesso em: 19 de junho de 2008.

MICROSOFT. **MSDN Library**. Disponível em: <http://msdn.microsoft.com/> Acesso em: 10 junho 2008.

MICROSOFT. **XNA Creators Club Online**. Disponível em: <http://creators.xna.com/> Acesso em: 10 junho 2008.