Uma biblioteca de Realidade Aumentada para a plataforma iOS

Acadêmico – Paulo Cesar Meurer
Orientador – Dalton Solano dos Reis

Roteiro

- Introdução
- Fundamentação teórica
- Desenvolvimento
- Resultados e discussão
- Conclusão / Extensão
- Demonstração

Introdução

- Objetivos do trabalho
  - Criar uma biblioteca de software que facilite a utilização dos recursos de câmera de vídeo, GPS, acelerômetro e bússola disponíveis na plataforma iOS, na criação de aplicações de RA
  - Criar uma camada de abstração que oculte detalhes da aquisição de imagens da câmera, do registro, do rastreamento, do ajuste visual dos objetos virtuais e da sobreposição das cenas do mundo real e virtual
  - Aplicar o conceito *markerless tracking*
  - Disponibilizar um aplicativo exemplo desenvolvido utilizando a biblioteca criada

FURB
UNIVERSIDADE REGIONAL DE BLUMENAU

Fundamentação teórica

- Realidade aumentada
  - Realidade e virtualidade
  - Interatividade em tempo real
  - Registro de objetos virtuais
    - Ponto de interesse (POI)
- Dispositivos
  - Dispositivos móveis

FURB
UNIVERSIDADE REGIONAL DE BLUMENAU

Fundamentação teórica

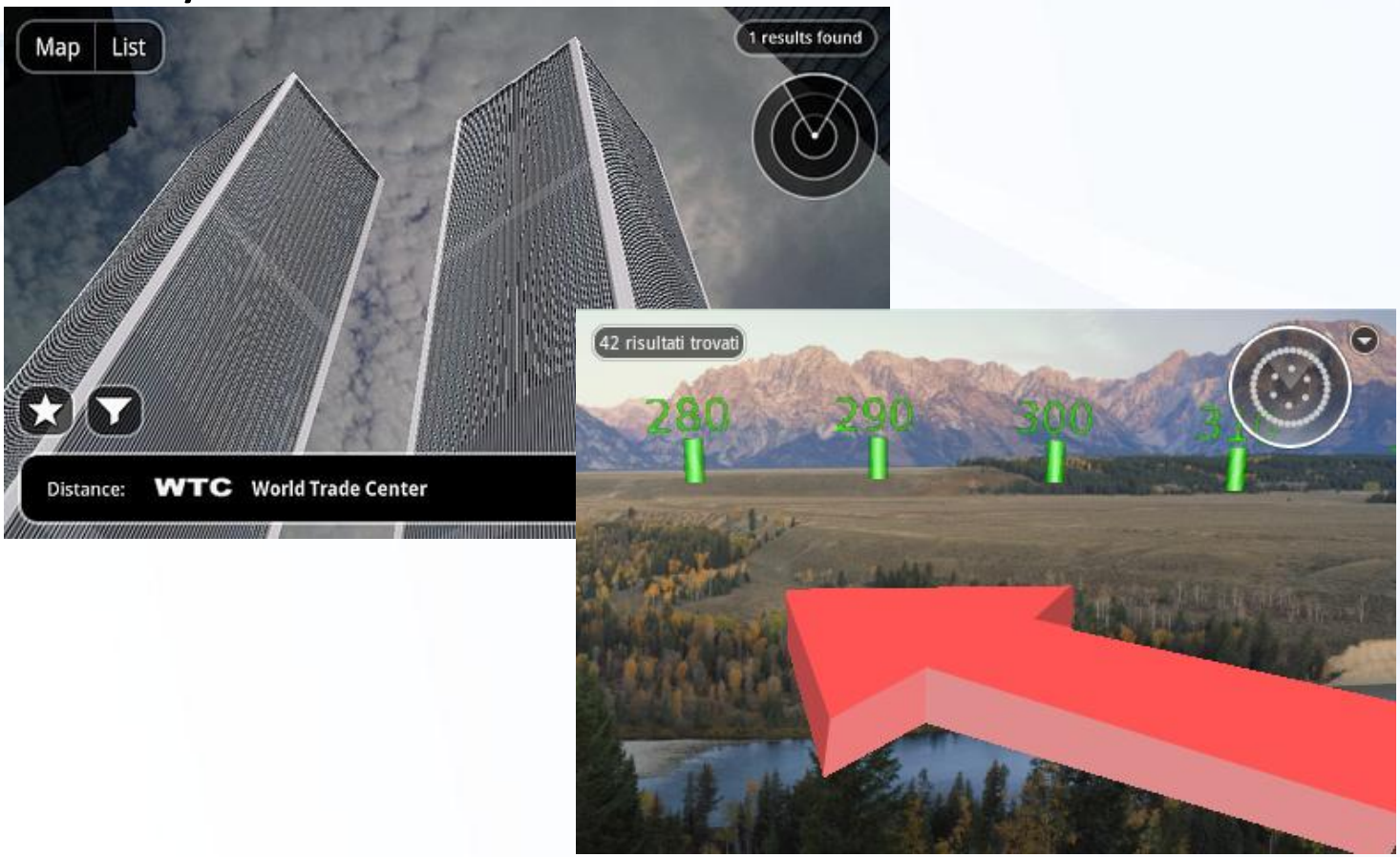
- Plataforma iOS
  - Darwin
  - Arquitetura
    - Cocoa Touch
    - Media
    - Core Services
    - Core OS
  - Localização e sensores
  - Câmera
  - OpenGL ES 1.1

Fundamentação teórica

- Trabalhos correlatos
  - Junaio

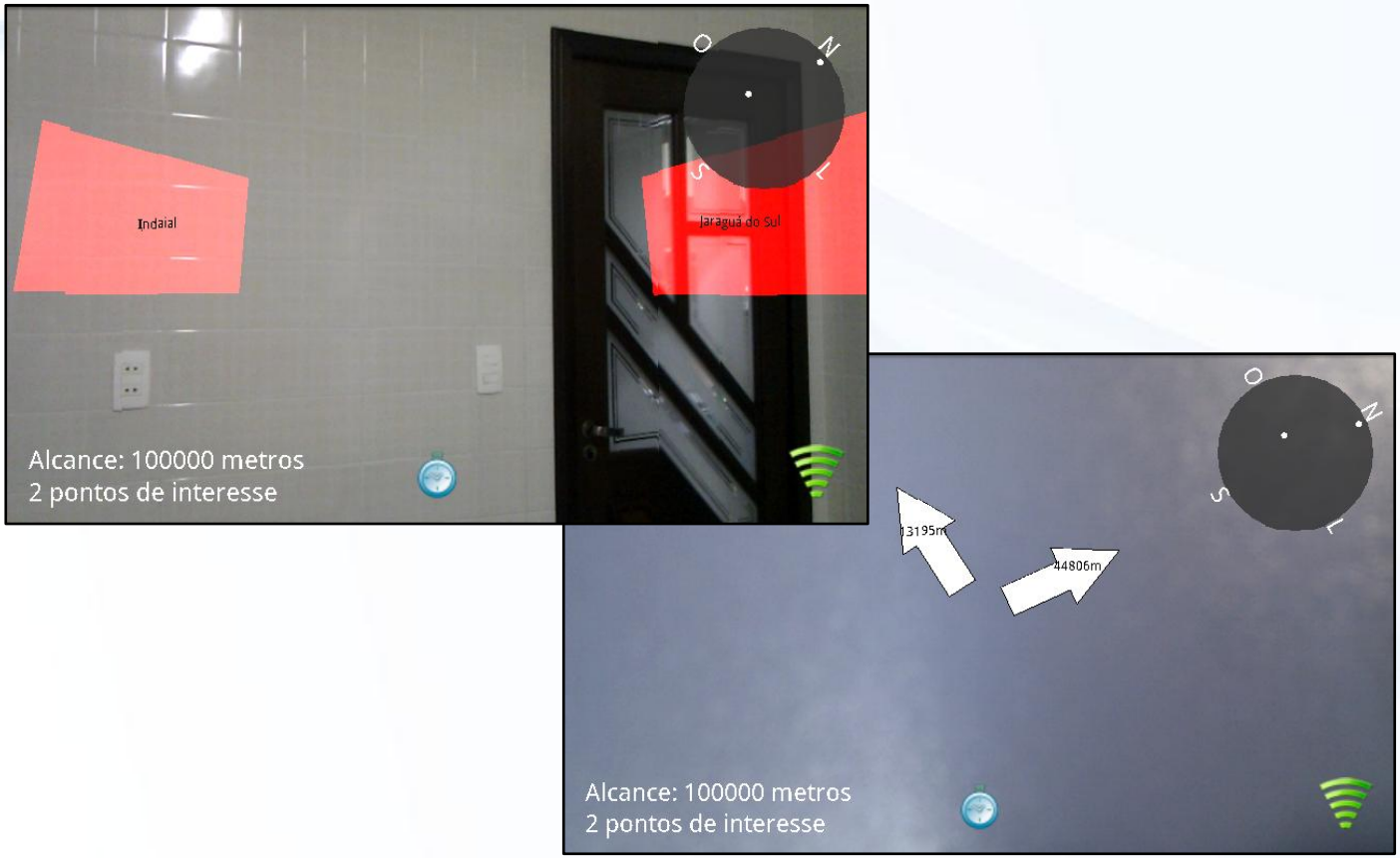
Fundamentação teórica

- Trabalhos correlatos
  - Layar



Fundamentação teórica

- Trabalhos correlatos
  - FURB Realidade Aumentada



Desenvolvimento

- Requisitos
  - Permitir visualizar o ambiente real através da câmera do dispositivo (RF)
  - Sobrepor ao ambiente real, objetos virtuais em 2D (RF)
  - Utilizar o multitoque para visualizar detalhes dos objetos virtuais (RF)
  - Utilizar o conceito *markerless tracking* para registro dos objetos virtuais através de coordenada geográfica(RF)

Desenvolvimento

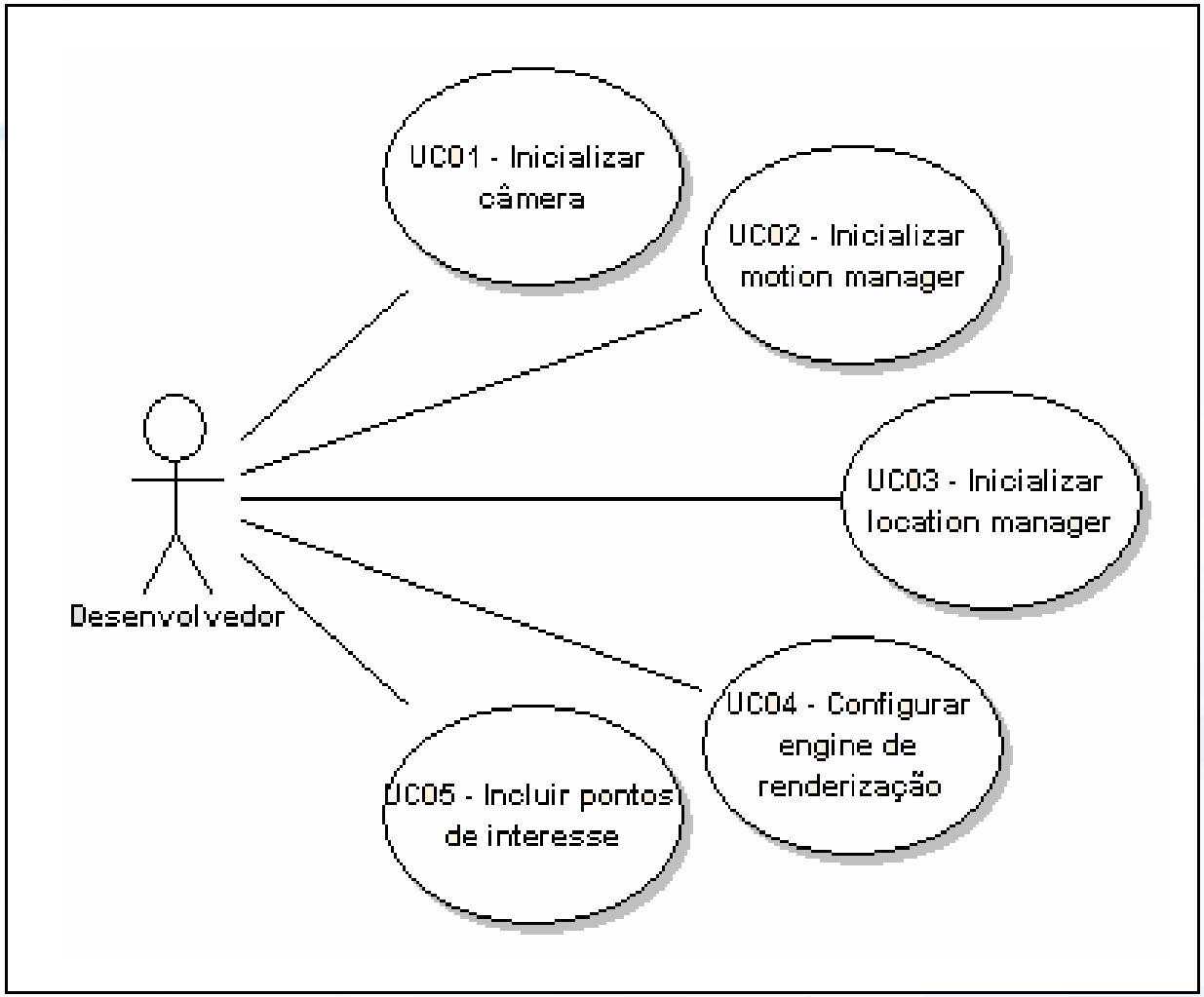
- Requisitos
  - Utilizar o acelerômetro para rastrear a direção que o usuário está visualizando com o dispositivo (RNF)
  - Ser implementado usando a plataforma iOS (RNF)

Desenvolvimento

- Especificação
  - Visão da biblioteca

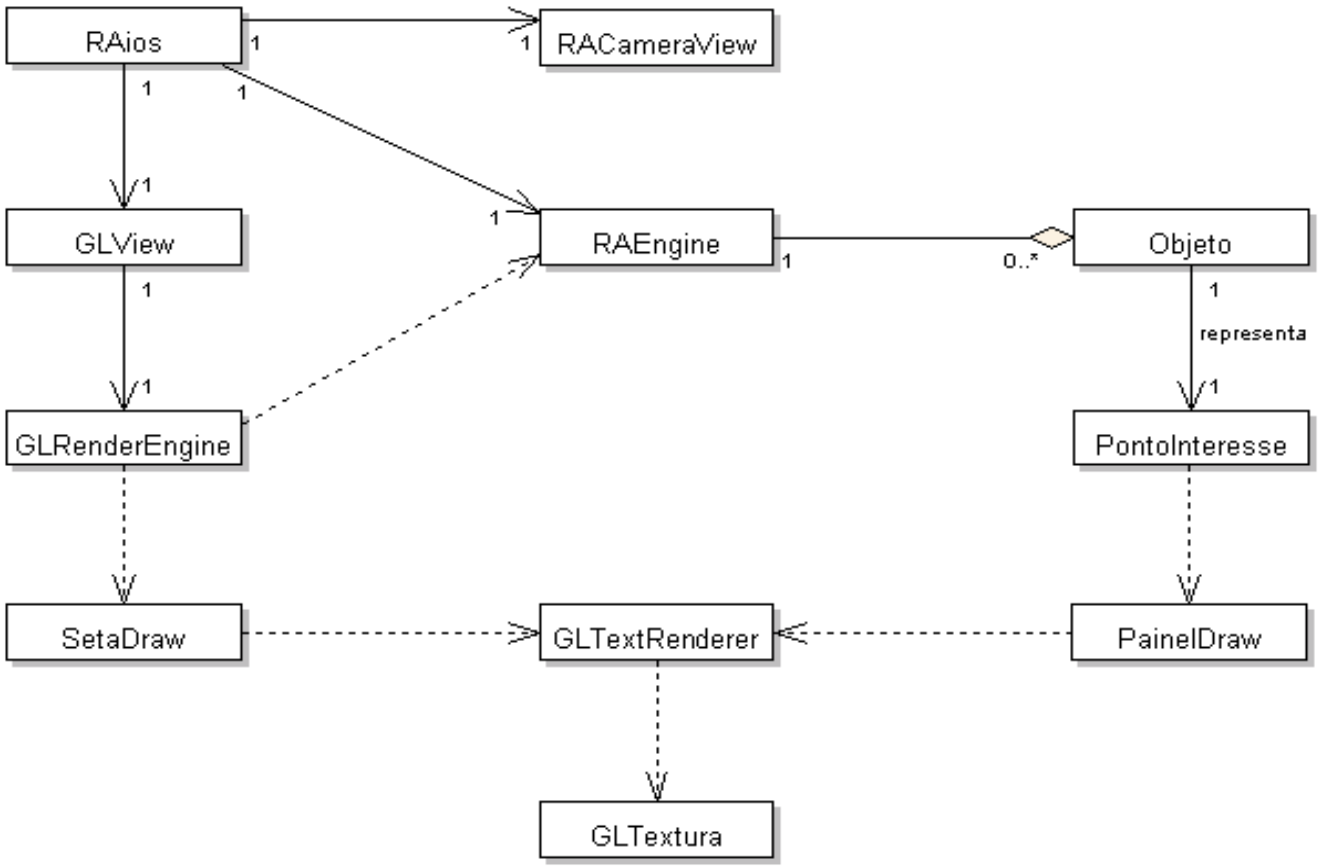
Desenvolvimento

- Especificação



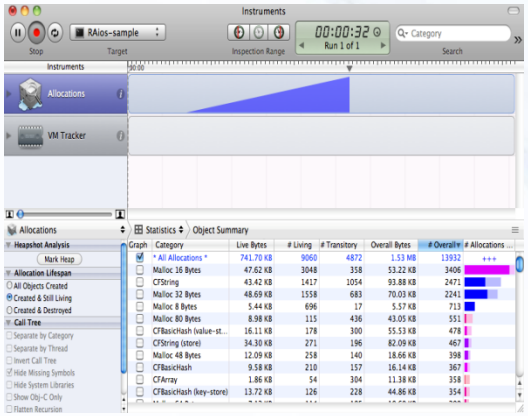
Desenvolvimento

- Principais classes



Desenvolvimento

- Técnicas e ferramentas utilizadas
  - Objective-C
  - Xcode
  - Instruments
  - iPhone 4



FURB
UNIVERSIDADE REGIONAL DE BLUMENAU

Desenvolvimento

- GLView
  - Cria o contexto para renderização do OpenGL ES
  - Sincroniza o desenho à taxa de atualização da tela
  - Não faz cálculos
  - Sensível ao toque

Desenvolvimento

- GLView

```
// cria o contexto de renderização e especifica qual versão do OpenGl vai ser utilizada
m_context = [[EAGLContext alloc]initWithAPI:kEAGLRenderingAPIOpenGLES1];

// seta o contexto EAGL como corrente. Isto significa que qualquer chamada ao OpenGL
nesta Thread vai ser amarrada a este contexto
if(!m_context || ![EAGLContext setCurrentContext:m_context]){
          [self release];
          return nil;
}
```

```
// cria o loop de animação
m_displayLink = [CADisplayLink displayLinkWithTarget:selfselector:@selector(drawView:)];
m_displayLink.frameInterval = 1;
[m_displayLink addToRunLoop:[NSRunLoop currentRunLoop]forMode:NSDefaultRunLoopMode];
```

Desenvolvimento

- Engine
  - Obtém valores do acelerômetro e da bússola
  - Obtém coordenadas geográficas do dispositivo
  - Calcula os pontos de interesse
    - Distância em metros
    - X, Y, Z e ângulo da seta e do objeto virtual
  - Atualizado pelas funções de callback dos sensores

Desenvolvimento

- Engine

```
// Guarda seno e conseno do ângulo.
double anguloSin = sin(rAngulo);
double anguloCos = cos(rAngulo);

// Guarda o ângulo pois vai usar na seta e no objeto.
obj->m_glAngulo = rAngulo * 180 / M_PI;

// Guarda a posição da seta no gl.
obj->m_glSetaX = (float) (glDiametroCamera * anguloSin);
obj->m_glSetaY = (float) (glDiametroCamera * anguloCos);
obj->m_glSetaZ = -8.0f;

// Guarda a posção do objeto no gl.
obj->m_glObjetoX = (float) (m_distanciaObjetoCamera * anguloSin);
obj->m_glObjetoY = (float) (m_distanciaObjetoCamera * anguloCos);
obj->m_glObjetoZ = -3.0f;
```

Desenvolvimento

- RenderEngine

```
// desenha os pontos de interesse
for (Objeto* obj in objects) {
          // verifica se o ponto está visivel
          if (obj->m_foraTela)
              continue;
          //(...)

          // alinha o ângulo e a posição para desenhar o objeto
          glTranslatef(obj->m_glObjetoX, obj->m_glObjetoY, obj->m_glObjetoZ);
          glRotatef(90.0, 1.0f, 0.0f, 0.0f);
          glRotatef(-obj->m_glAngulo, 0.0f, 1.0f, 0.0f);

          [obj onDraw];

          //(...)
}
```

Desenvolvimento

- Operacionalidade da biblioteca

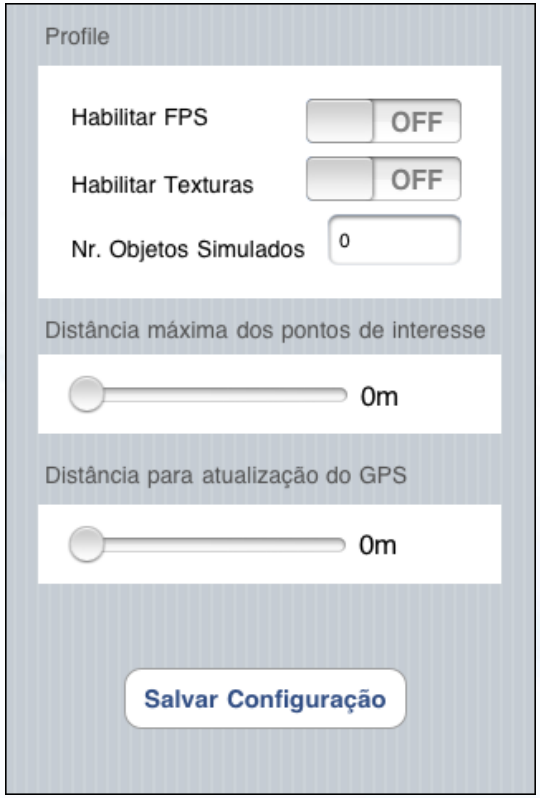
```
// Inicializa a engine
m_RAios = [[RAios alloc]init];
```

```
[m_RAios arInit:self.view];   // Inicializa a biblioteca

[m_RAios arVideoCapStart];    // inicia a captura de vídeo

[m_RAios arMotionStart];      // inicia a captura de movimento

[m_RAios arLocationStart];    // inicia a captura das coordenadas GPS

// adiciona um ponto de interesse
PontoInteresse *shopping = [[PontoInteresse alloc]initWithDescricao:
               @"Shopping" latitude:-26.919700 longitude:-49.069400];

[m_RAios arInsertObject:shopping];
[shopping release];

[m_RAios arDrawStart];        // inicia o desenho dos objetos
```

FURB
UNIVERSIDADE REGIONAL DE BLUMENAU

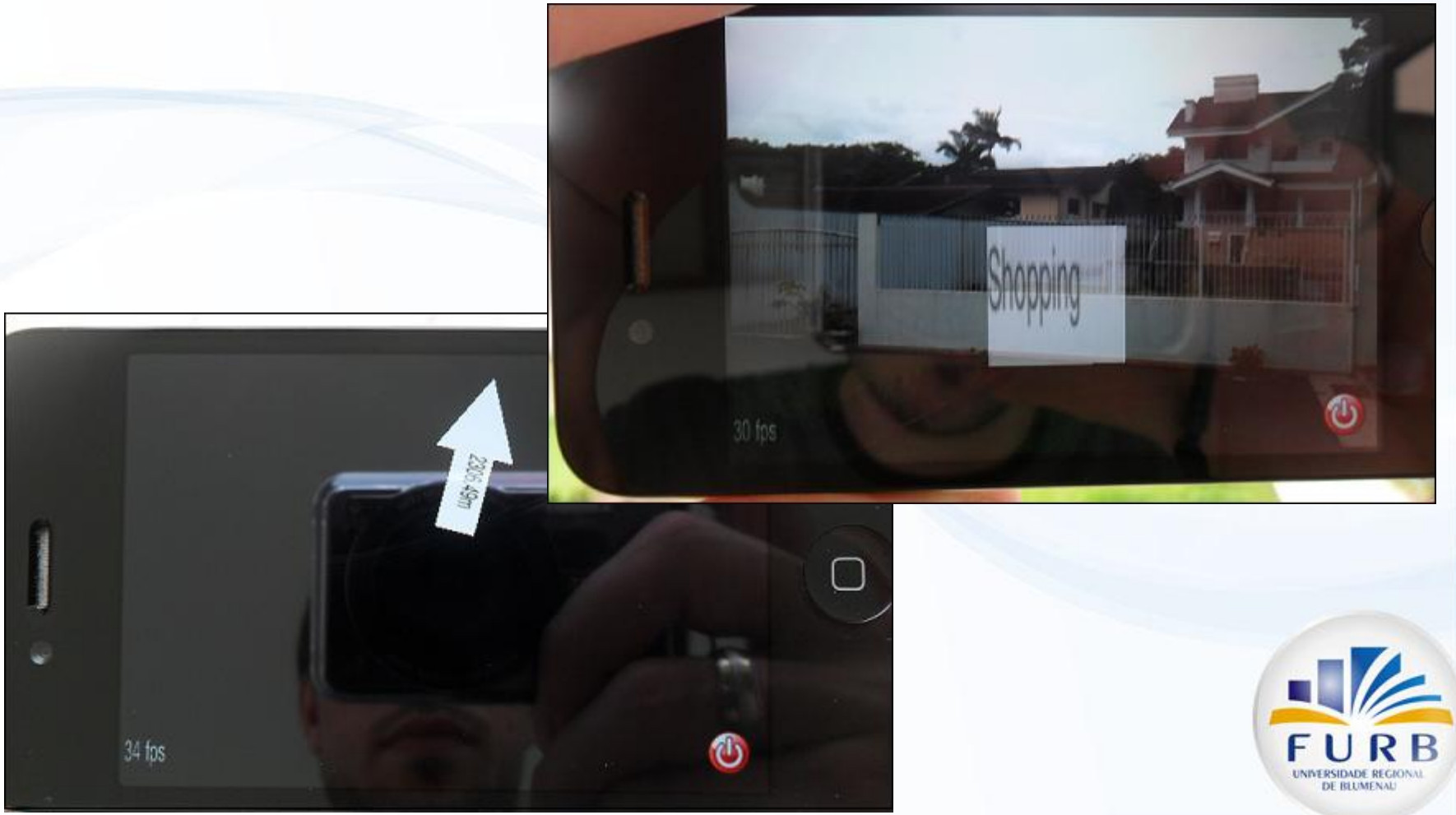
Desenvolvimento

- Operacionalidade da aplicação



Desenvolvimento

- Operacionalidade da aplicação



Resultados e discussão

- Realidade aumentada
  - Realidade e virtualidade através da câmera e OpenGL ES
  - Registro dos objetos virtuais pela coordenada geográfica
  - Interatividade pela bússola e acelerômetro
- OpenGL ES
  - Desenho de texto
- Biblioteca
  - Êxito na criação de uma camada de abstração
  - Disponibilizada aplicação

FURB
UNIVERSIDADE REGIONAL DE BLUMENAU

Resultados e discussão

- Medições do FPS para conteúdo transparente

Array de vértices

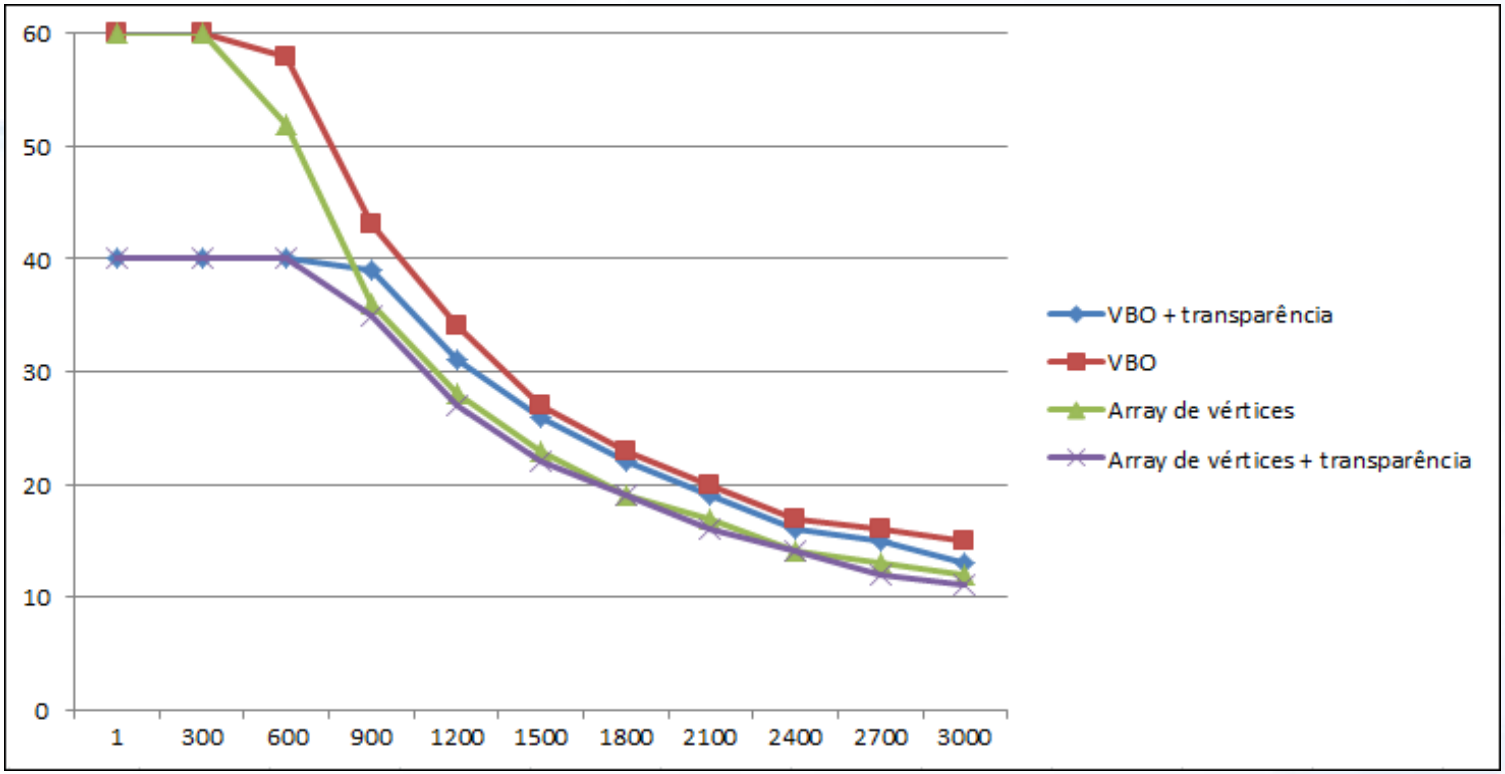
Quantidade de objetos	Média FPS sem transparência	Média FPS com transparência
1	60	40
300	60	40
600	52	40
900	36	35
1200	28	27
1500	23	22
1800	19	19
2100	17	16
2400	14	14
2700	13	12
3000	12	11

VBO

Quantidade de objetos	Média FPS sem transparência	Média FPS com transparência
1	60	40
300	60	40
600	58	40
900	43	39
1200	34	31
1500	27	26
1800	23	22
2100	20	19
2400	17	16
2700	16	15
3000	15	13

Resultados e discussão

- Gráfico do impacto da transparência



Resultados e discussão

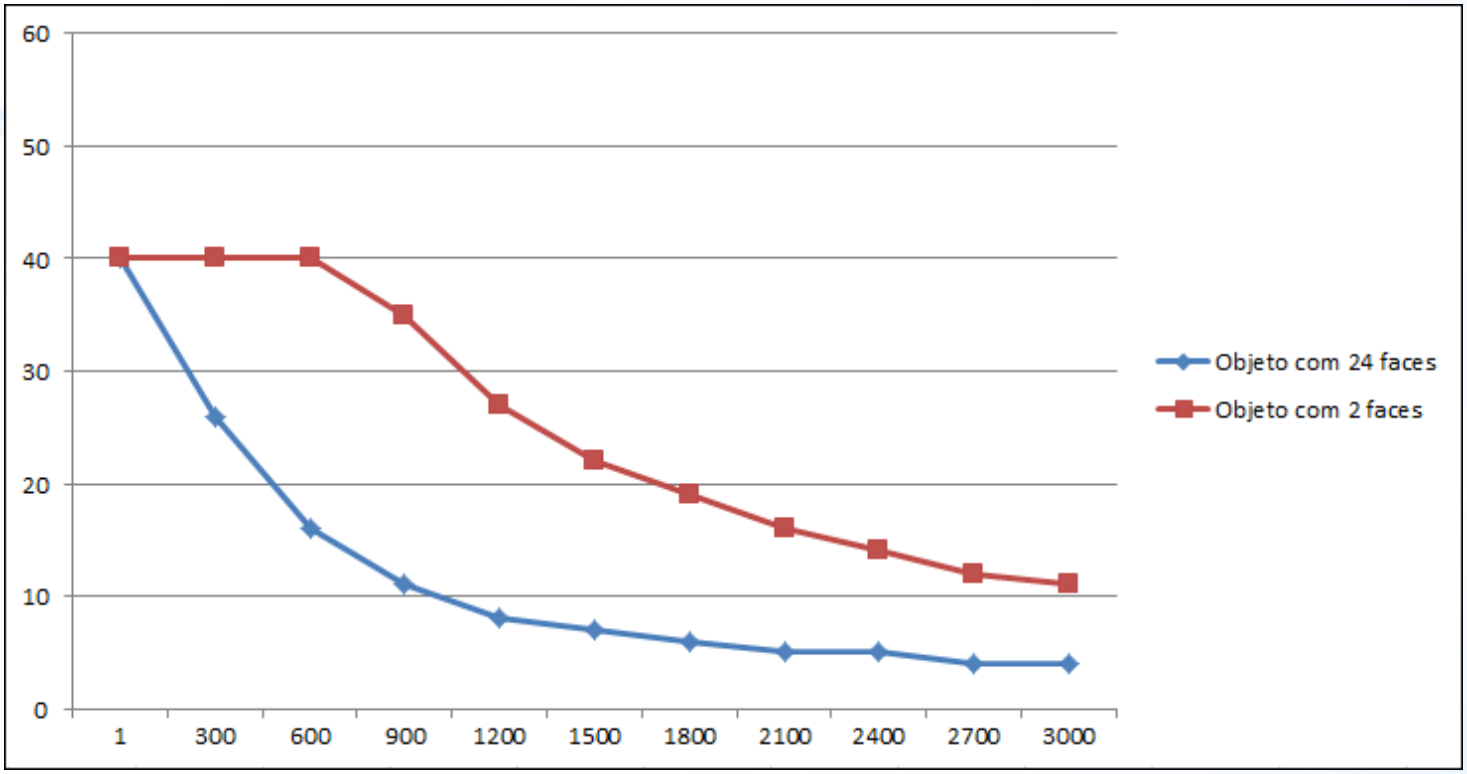
- Medições do FPS para complexidade do objeto

Quantidade de objetos	Média FPS com 2 faces	Média FPS com 24 faces
1	40	40
300	40	26
600	40	16
900	35	11
1200	27	8
1500	22	7
1800	19	6
2100	16	5
2400	14	5
2700	12	4
3000	11	4

FURB
UNIVERSIDADE REGIONAL DE BLUMENAU

Resultados e discussão

- Gráfico do desempenho para complexidade



Resultados e discussão

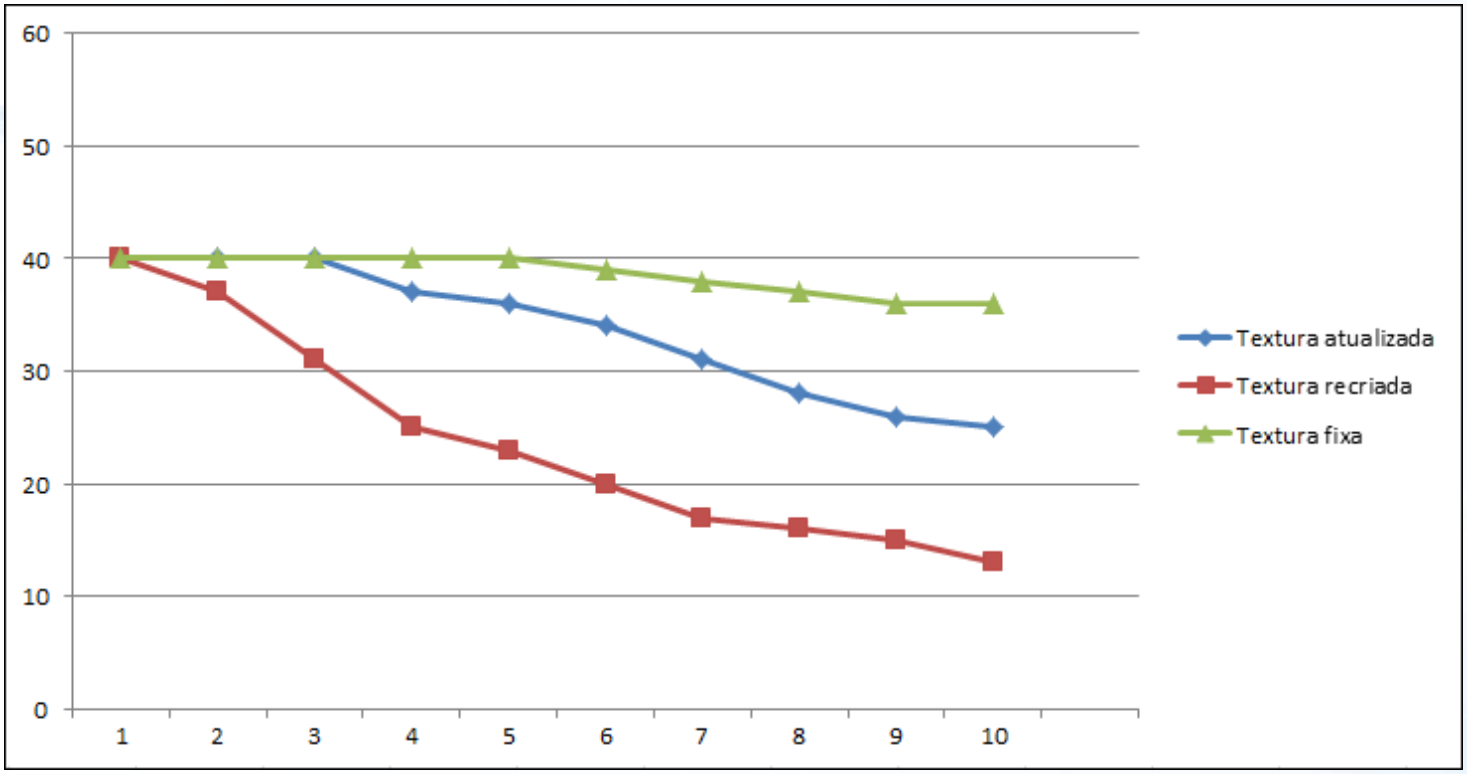
- Medições do FPS para o desenho de texturas

Quantidade de objetos	Média FPS recriando a textura	Média FPS atualizando a textura	Média FPS com textura fixa
1	40	40	40
2	37	40	40
3	31	40	40
4	25	37	40
5	23	36	40
6	20	34	39
7	17	31	38
8	16	28	37
9	15	26	36
10	13	25	36

FURB
UNIVERSIDADE REGIONAL DE BLUMENAU

Resultados e discussão

- Gráfico do desempenho para texturas



Conclusão

- Biblioteca
- Aplicação
- Plataforma iOS
  - Xcode
  - Instruments
  - Simulador
- Semelhança com os correlatos
- Interface simples

FURB
UNIVERSIDADE REGIONAL DE BLUMENAU

Extensões

- Desenhar objetos virtuais
  - VBO
  - Apenas objetos no campo de visão
- OpenGL ES 2.0
- glDrawArrays
- Texturas
  - Atlas
- Oclusão de objetos
- Suporte à mudança de orientação

Extensões

- Radar
- Marcadores fixos
- Aquisição de pontos de interesse
- Giroscópio

Demonstração

Demonstração prática

Obrigado