VISEDU-CG 3.0: Aplicação didática para visualizar material educacional – Módulo de Computação Gráfica

SAMUEL ANDERSON NUNES

ORIENTADOR: DALTON SOLANO DOS REIS

FURB – Universidade Regional de Blumenau
DSC – Departamento de Sistemas e Computação
Grupo de Pesquisa em Computação Gráfica, Processamento de Imagens e Entretenimento Digital
www.inf.furb.br/gcg

Roteiro

- Introdução
- Objetivos
- Fundamentação Teórica
- Desenvolvimento
- Resultados e Discussões
- Conclusões e Extensões

Introdução

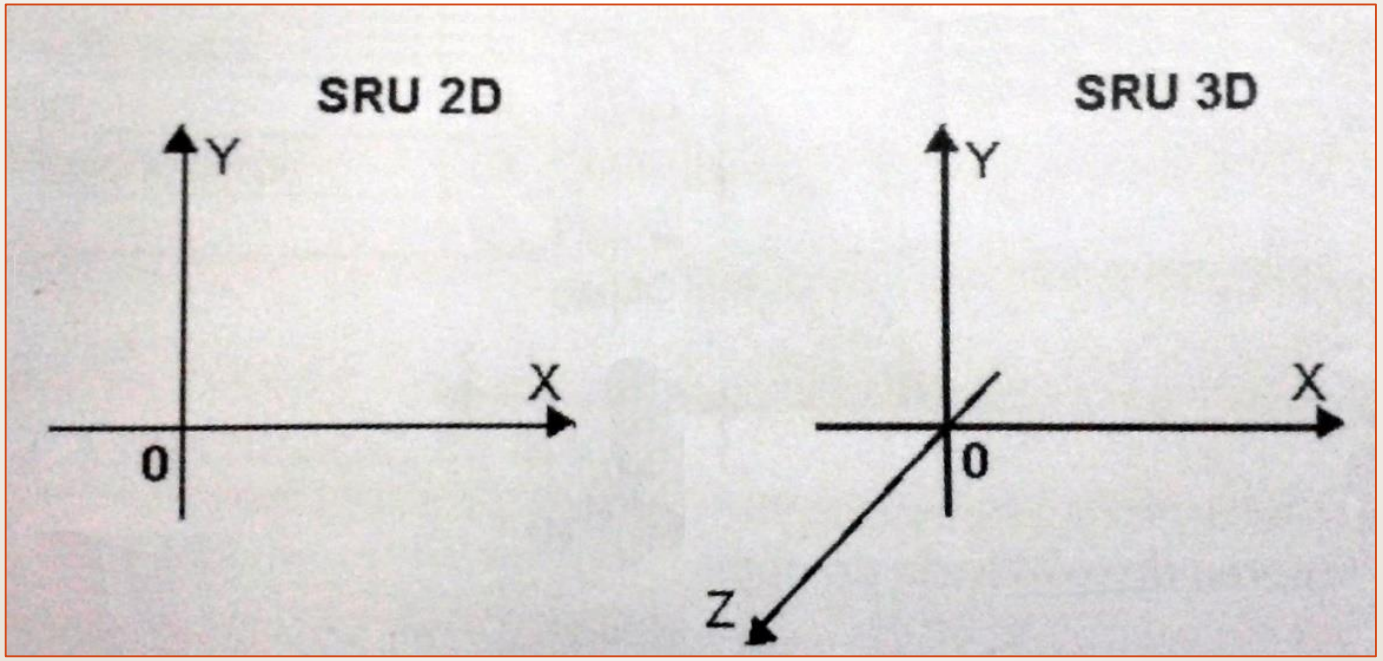
- O crescimento da Internet
- Evolução dos navegadores
- Computação gráfica na Internet
- A educação através da Internet
- Software educacional
  - Tecnologias Gráficas
  - Independência de plataforma
- VisEdu-CG (MONTIBELER, 2014) – Visualizador de material educacional

Objetivos

- Incluir o tipo de objeto Polígono e Spline 3D
- Adicionar na representação gráfica o uso da Iluminação
- Disponibilizar a seleção de objetos no espaço 3D
- Disponibilizar as mesmas funcionalidades gráficas em 3D num espaço 2D

Fundamentação Teórica
Representação do Espaço Gráfico

- Representação do Espaço Gráfico
  - Sistema de Referência do Universo (SRU)
  - Sistema de coordenadas cartesianas

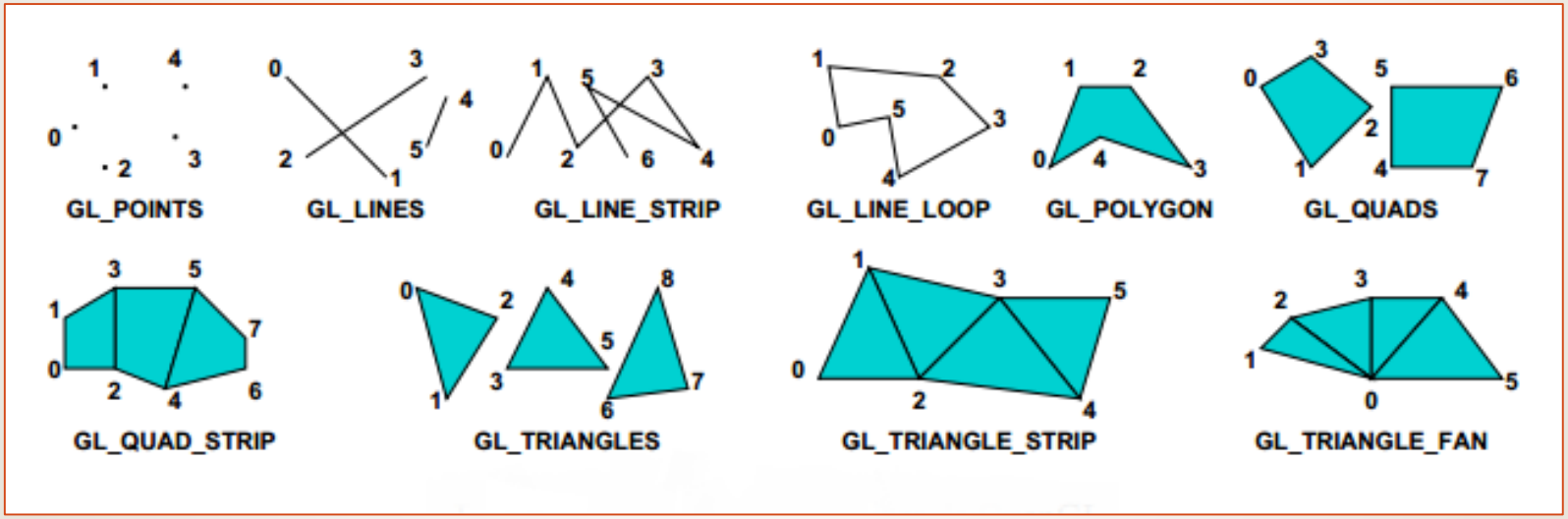


Fundamentação Teórica
Representação dos objetos gráficos

- Representação da geometria (forma) e atributos (propriedades) de um objeto do mundo real (MONTENEGRO, 2013)
- Definidos através da especificação de seus atributos (CATARINA, 2013)
  - Geométricos: Métricas do objeto
  - Topológicos: Forma dos objetos
  - Cor: informações relacionadas a textura dos objetos

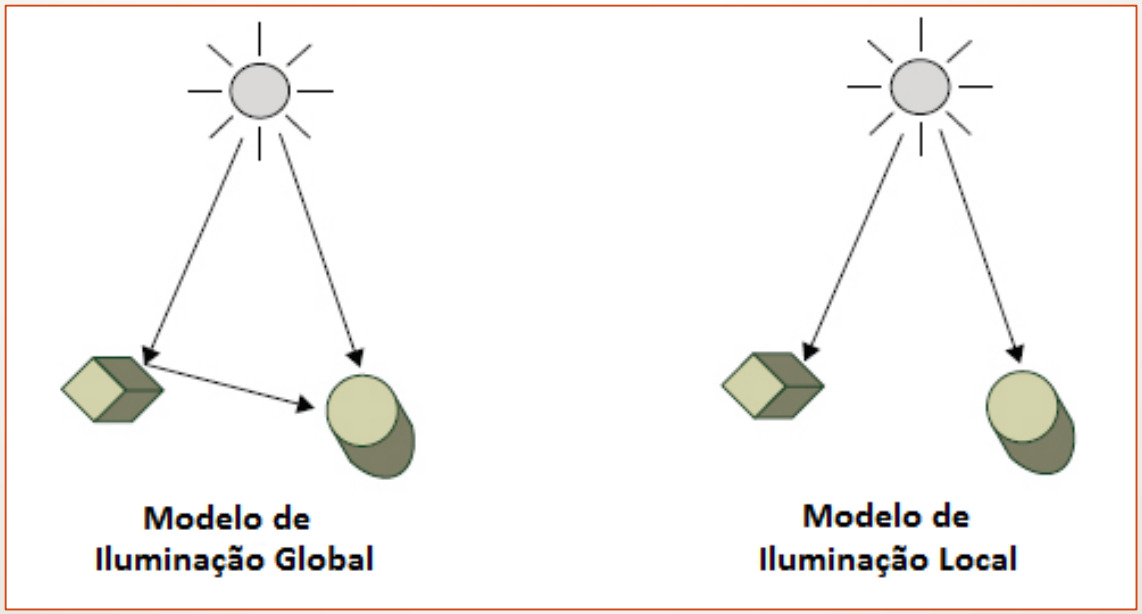
Fundamentação Teórica
Representação dos objetos gráficos

- Primitivas gráficas
  - Construída a partir dos vértices do objeto gráfico



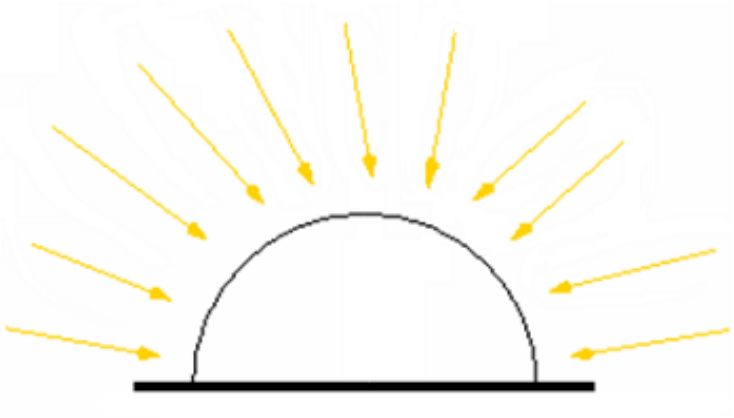
Fundamentação Teórica
Iluminação

- **Modelos de iluminação**
  - **Global**: Utiliza informação de outros objetos que estão sendo iluminados diretamente.
  - **Local**: Representa apenas a luz que vem de fontes diretas.

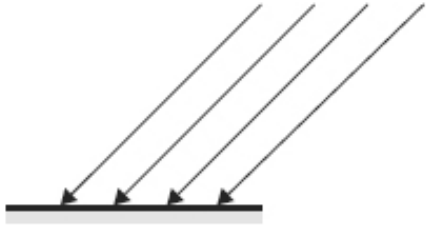


Fundamentação Teórica
Iluminação

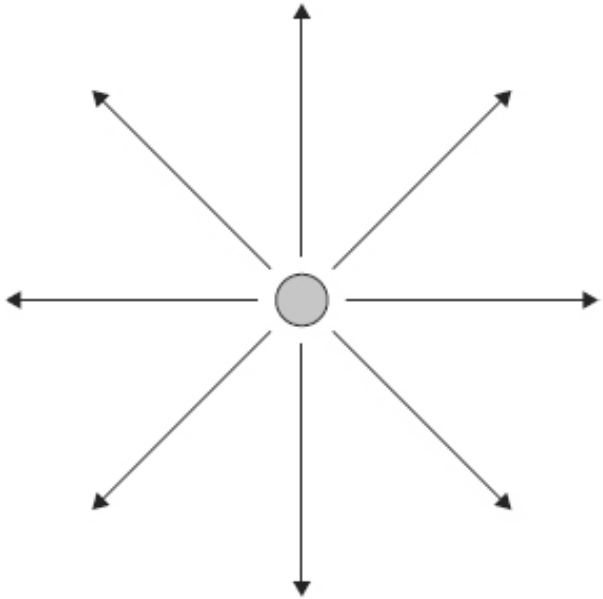
- **Tipos de luzes**
  - Luz Ambiente (Ambient Light)
  - Ponto de Luz (Point Light)
  - Luz Direcional (Direcional Light)
  - Luz Refletora (Spot Light)



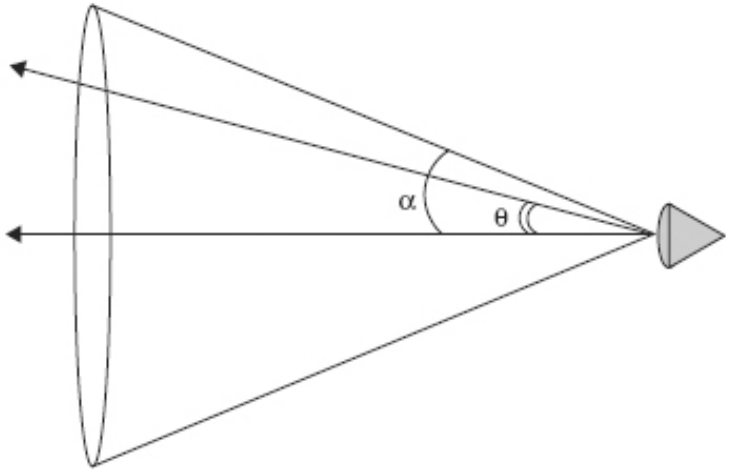
Luz Ambiente (AmbientLight)



Luz Direcional (DirectionalLight)



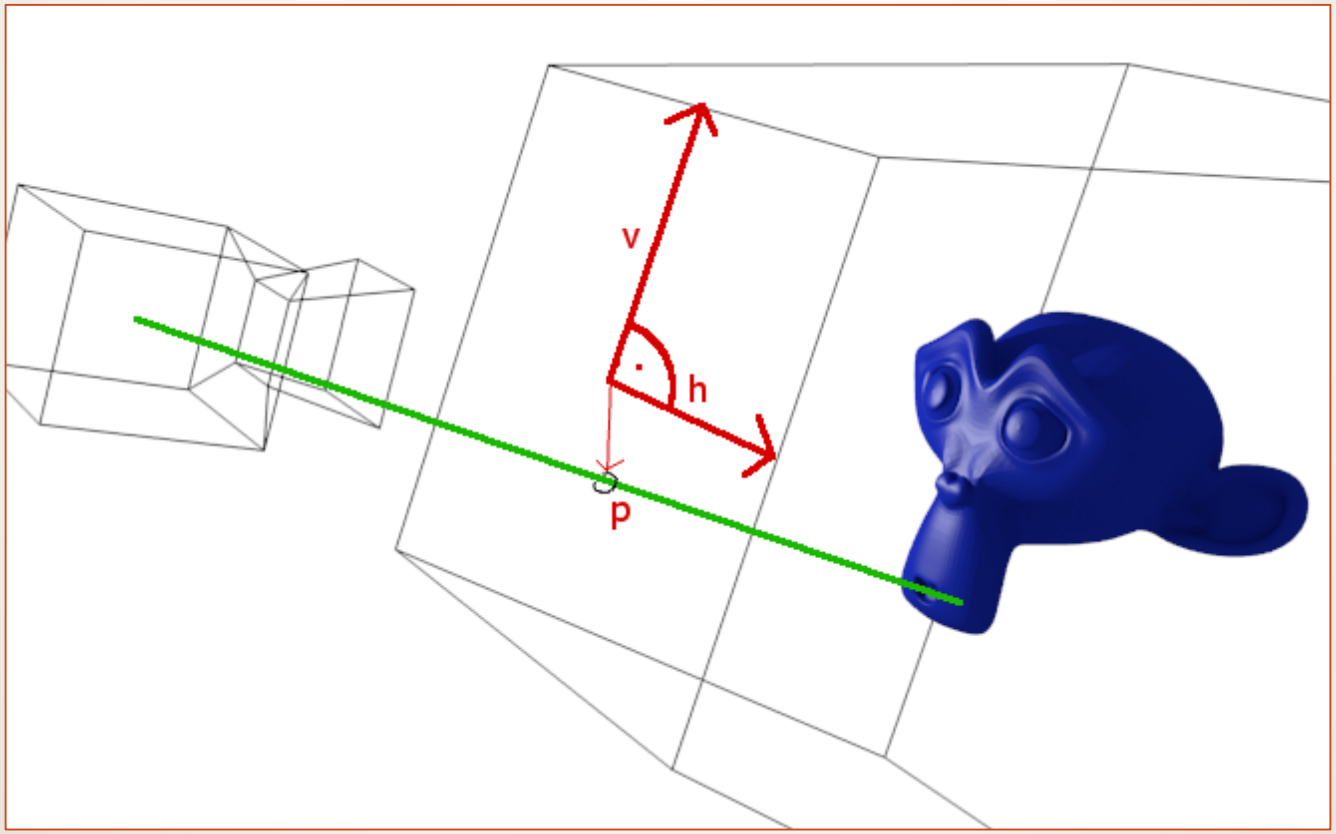
Ponto Luz (PointLight)



Luz Refletora (SpotLight)

Fundamentação Teórica
Seleção em ambiente 3D

- **Algoritmo *Ray Picking***



Fundamentação Teórica
HTML5 e WebGL

- **HTML5**
  - Semântica e acessibilidade
  - Novos recursos (Canvas, reprodução de áudio e vídeo, WebGL)
- **WebGL**
  - JavaScript
  - Baseada na OpenGL ES 2.0
  - Funcionalidade de renderização para um contexto HTML
  - **Three.js**
    - Biblioteca gráfica 3D JavaScript de código aberto
    - Suporte completo para as funcionalidades da WebGL

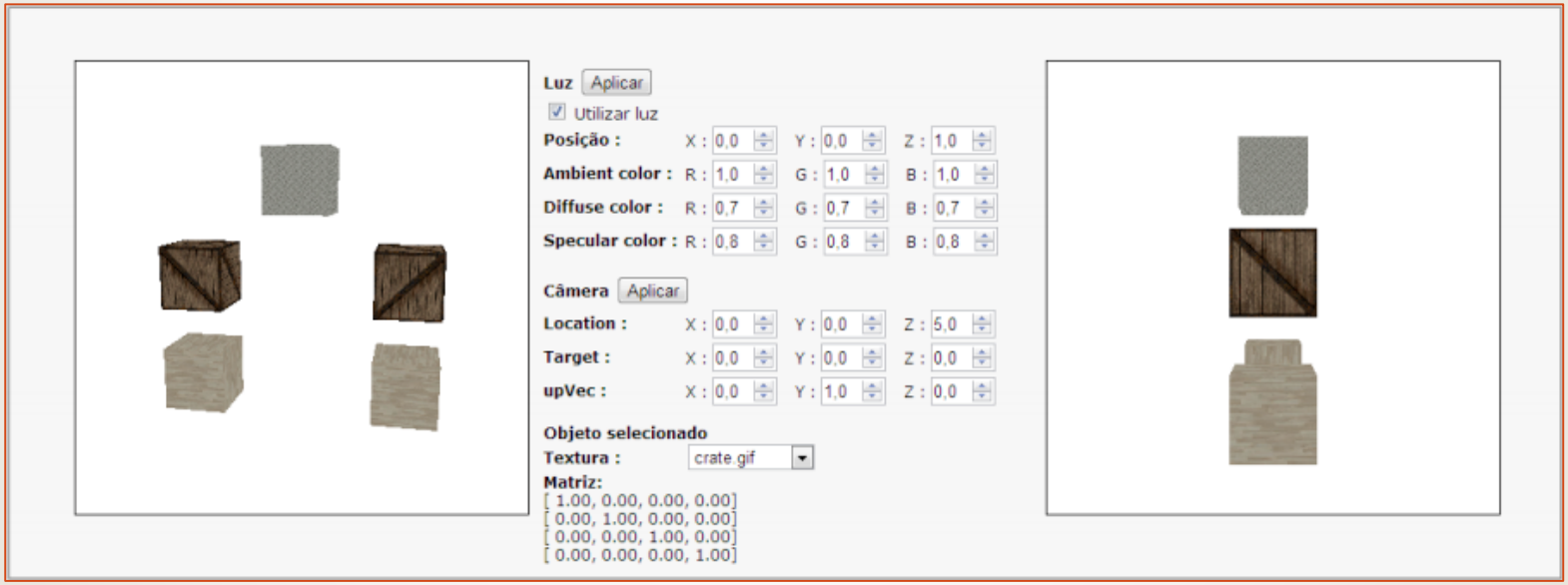
Fundamentação Teórica
Trabalhos correlatos

- **StarLogo TNG**
  - Desenvolvido pelo MIT
  - Motivar pessoas mais jovens na programação por meio de ferramentas que facilitam o desenvolvimento de sistemas
  - Suporte a gráficos 3D
  - Recursos de áudio

Fundamentação Teórica
Trabalhos correlatos

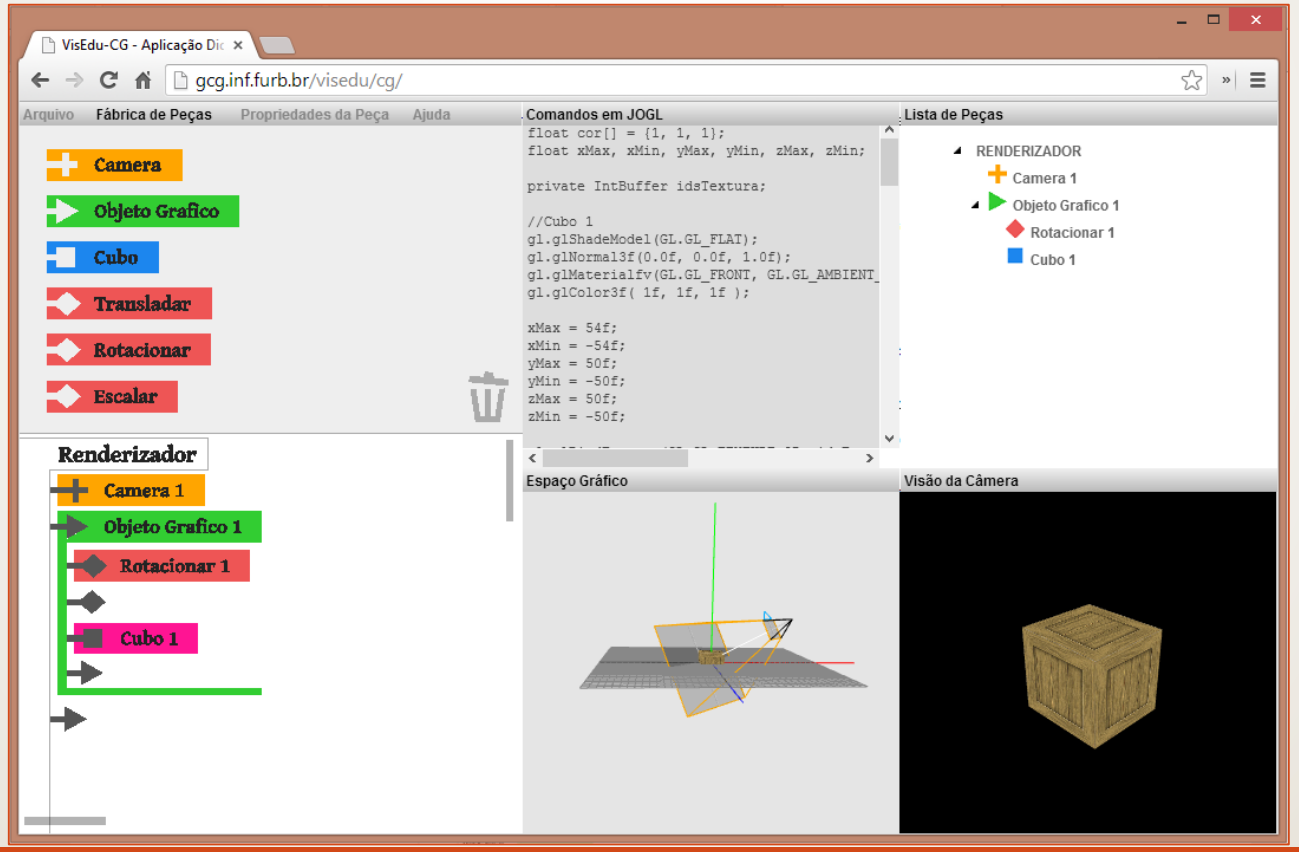
Fundamentação Teórica
Trabalhos correlatos

- **Motor de jogos 3D** (PEREIRA, 2012)



Fundamentação Teórica
Trabalhos correlatos

- **VisEdu 2.0** (Montibeler, 2014)



Desenvolvimento
Requisitos

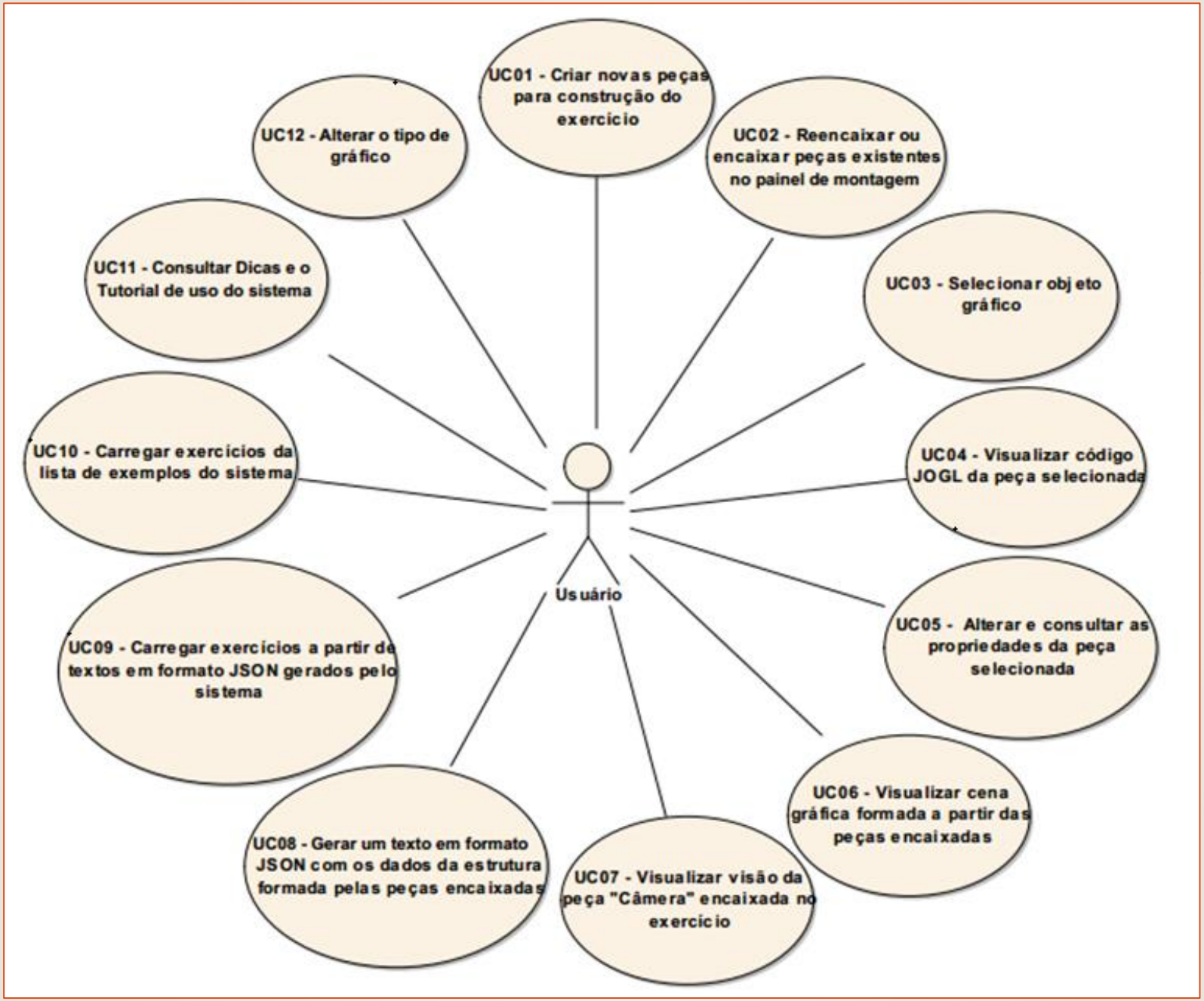
- **Requisitos Funcionais**
  - Permitir optar por trabalhar em 2D ou 3D
  - Permitir mudar a primitiva gráfica e o tipo de desenho utilizado
  - Permitir manipular os controles de iluminação da cena
  - Permitir incluir tipo de objeto Polígono e Spline 3D
  - Permitir selecionar os objetos gráficos no espaço 3D

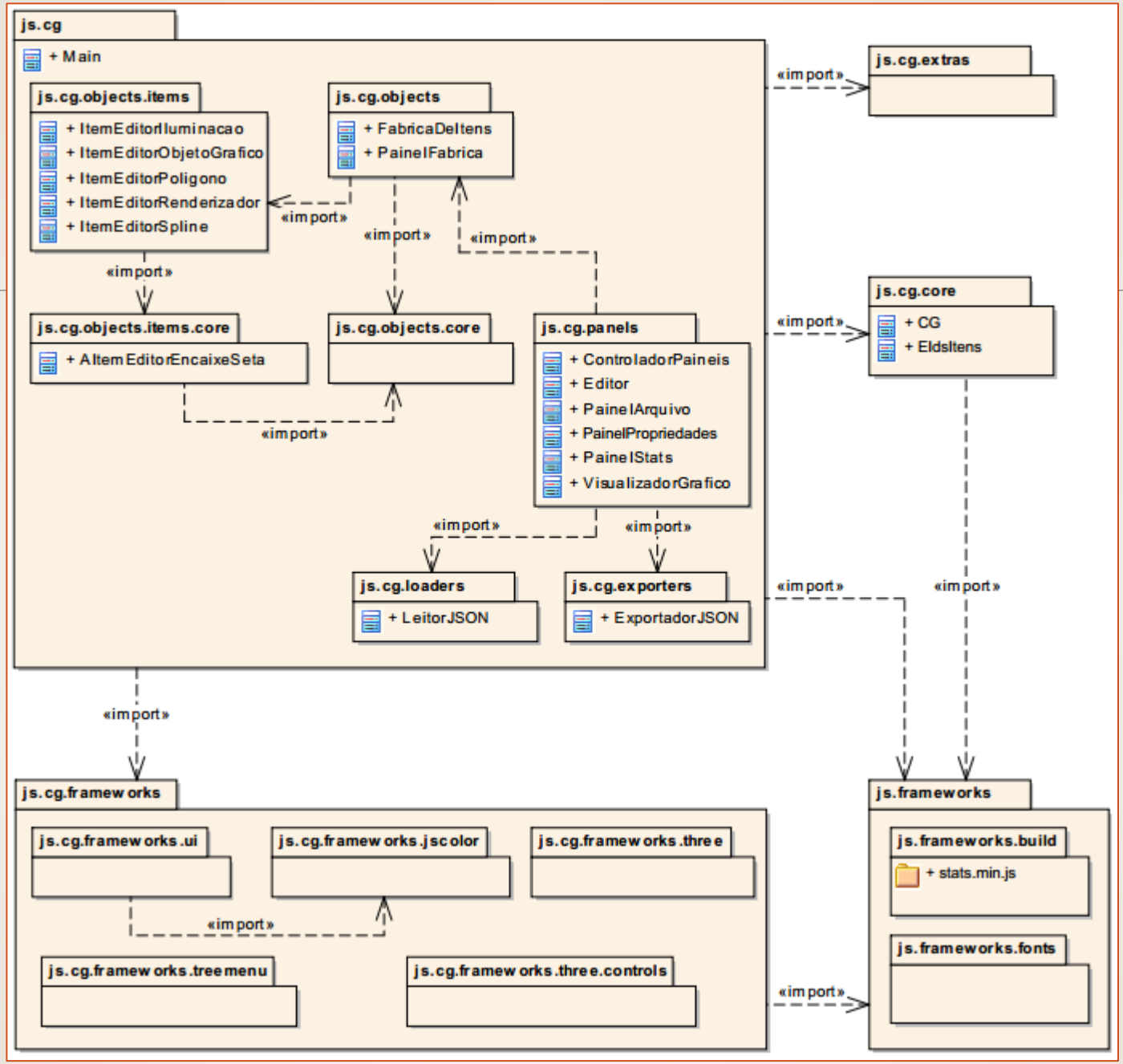
Desenvolvimento
Requisitos

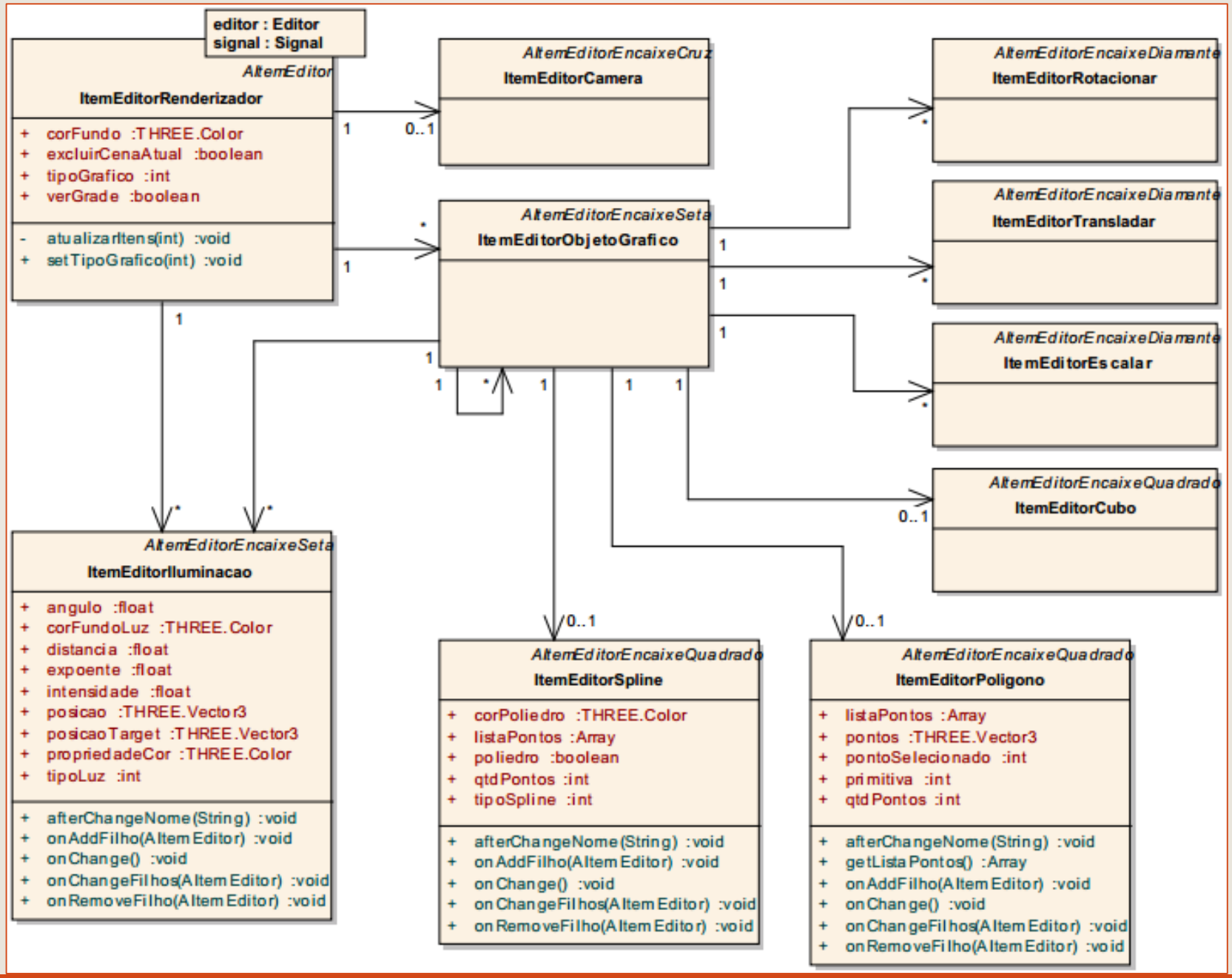
- **Requisitos não Funcionais**
  - Ser desenvolvido na linguagem HTML5
  - Utilizar o WebGL para executar as funções de computação gráfica
  - Ser de código aberto

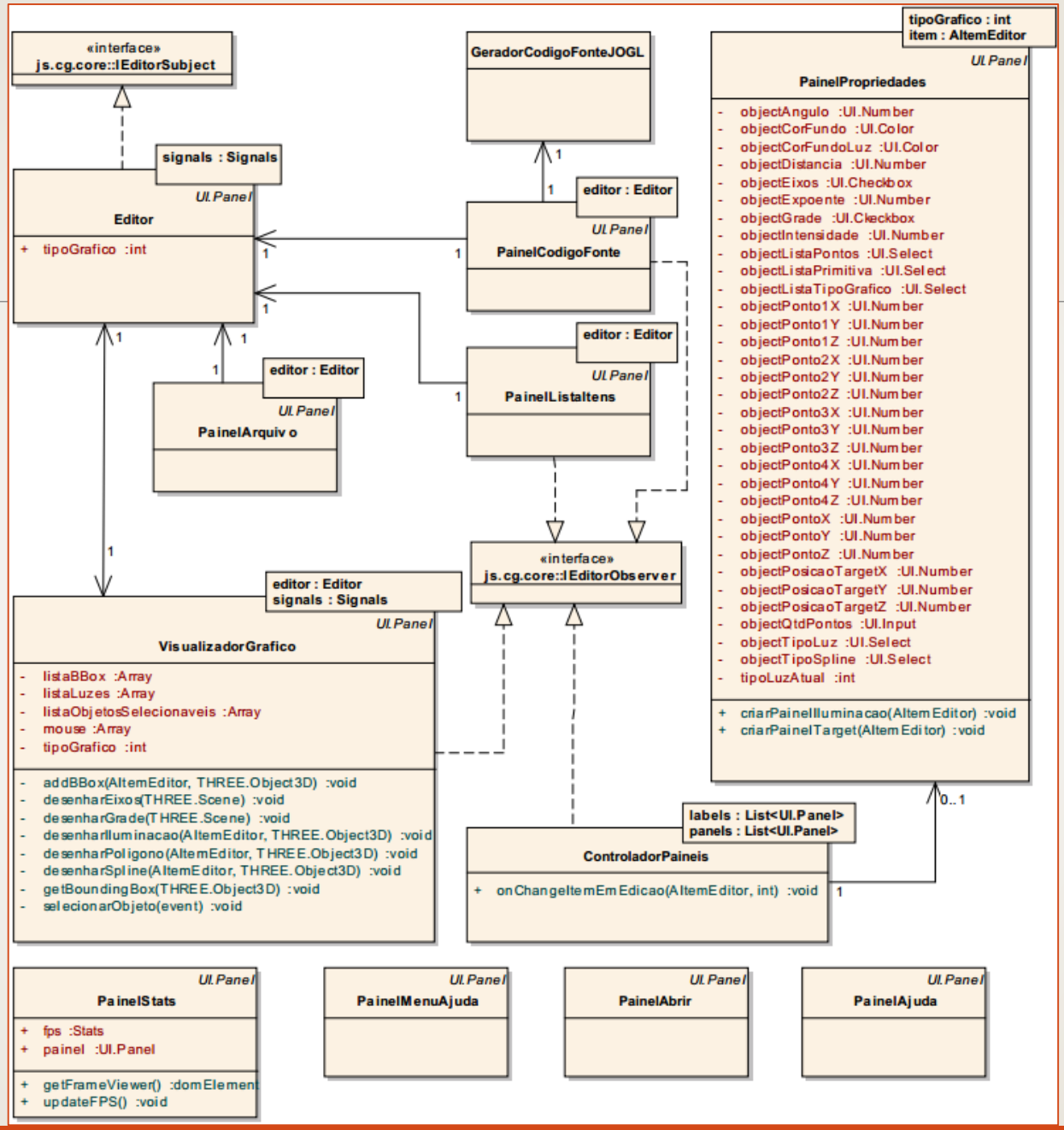
Desenvolvimento
Especificação

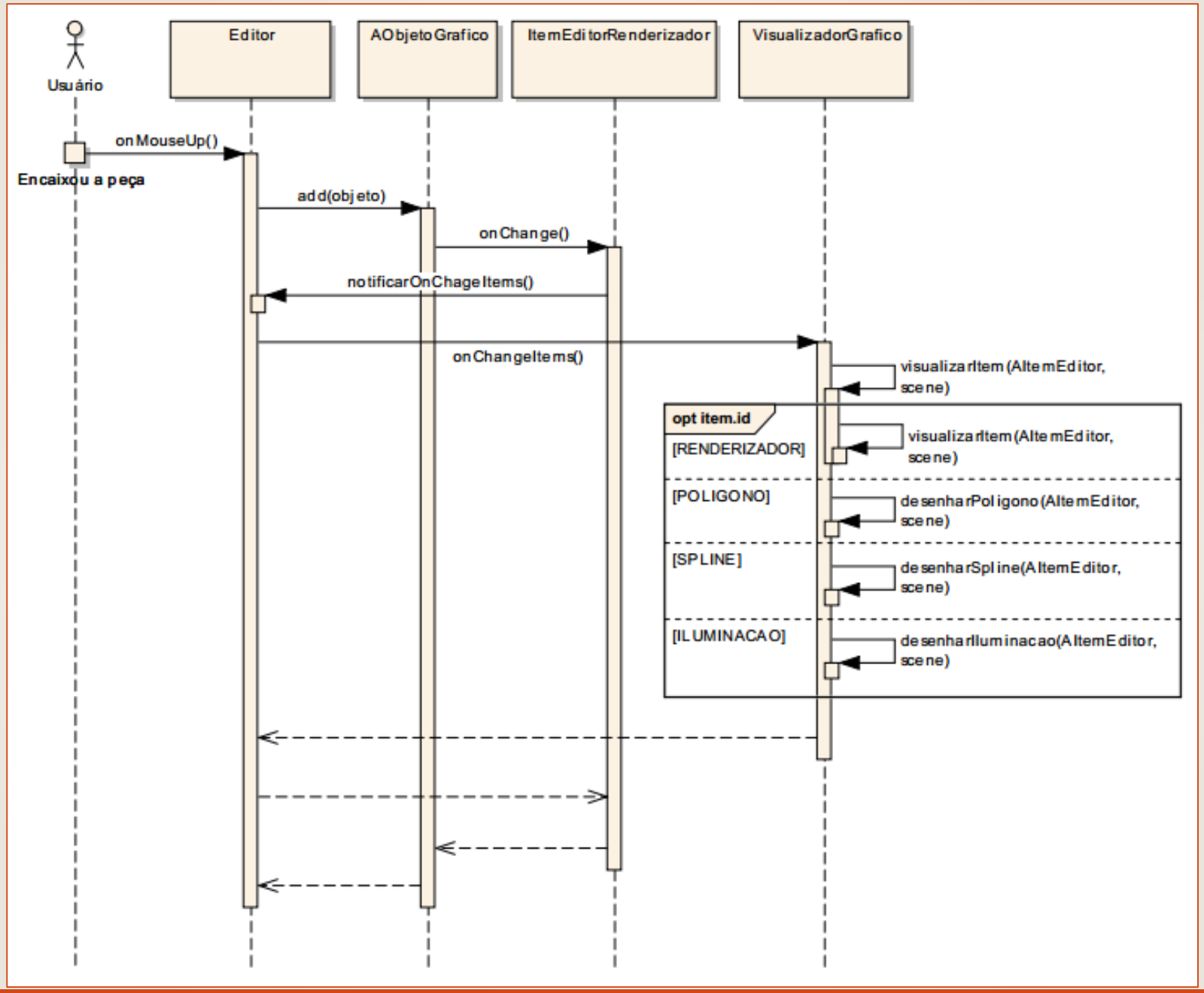
- Diagrama de casos de uso
- Diagrama de pacotes/classes
- Diagrama de sequencia


UC01 - Criar novas peças para construção do exercício
UC02 - Reencaixar ou encaixar peças existentes no painel de montagem
UC03 - Selecionar objeto gráfico
UC04 - Visualizar código JOGL da peça selecionada
UC05 - Alterar e consultar as propriedades da peça selecionada
UC06 - Visualizar cena gráfica formada a partir das peças encaixadas
UC07 - Visualizar visão da peça "Câmera" encaixada no exercício
UC08 - Gerar um texto em formato JSON com os dados da estrutura formada pelas peças encaixadas
UC09 - Carregar exercícios a partir de textos em formato JSON gerados pelo sistema
UC10 - Carregar exercícios da lista de exemplos do sistema
UC11 - Consultar Dicas e o Tutorial de uso do sistema
UC12 - Alterar o tipo de gráfico
Usuário


js.cg
+ Main
js.cg.objects.items
+ ItemEditorIluminacao
+ ItemEditorObjetoGrafico
+ ItemEditorPoligono
+ ItemEditorRenderizador
+ ItemEditorSpline
js.cg.objects
+ FabricaDeItens
+ PainelFabrica
«import»
js.cg.extras
js.cg.objects.items.core
+ AItemEditorEncaixeSeta
js.cg.objects.core
js.cg.panels
+ ControladorPaineis
+ Editor
+ PainelArquivo
+ PainelPropriedades
+ PainelStats
+ VisualizadorGrafico
js.cg.core
+ CG
+ EIdsItens
js.cg.loaders
+ LeitorJSON
js.cg.exporters
+ ExportadorJSON
js.cg.frameworks
js.cg.frameworks.ui
js.cg.frameworks.jscolor
js.cg.frameworks.three
js.cg.frameworks.treemenu
js.cg.frameworks.three.controls
js.frameworks
js.frameworks.build
+ stats.min.js
js.frameworks.fonts


editor : Editor
signal : Signal
AItemEditor
ItemEditorRenderizador
+ corFundo :THREE.Color
+ excluirCenaAtual :boolean
+ tipoGrafico :int
+ verGrade :boolean
- atualizarItens(int) :void
+ setTipoGrafico(int) :void
AItemEditorEncaixeCruz
ItemEditorCamera
AItemEditorEncaixeDiamante
ItemEditorRotacionar
AItemEditorEncaixeDiamante
ItemEditorTransladar
AItemEditorEncaixeDiamante
ItemEditorEscalar
AItemEditorEncaixeSeta
ItemEditorObjetoGrafico
AItemEditorEncaixeQuadrado
ItemEditorCubo
AItemEditorEncaixeSeta
ItemEditorIluminacao
+ angulo :float
+ corFundoLuz :THREE.Color
+ distancia :float
+ expoente :float
+ intensidade :float
+ posicao :THREE.Vector3
+ posicaoTarget :THREE.Vector3
+ propriedadeCor :THREE.Color
+ tipoLuz :int
+ afterChangeNome(String) :void
+ onAddFilho(AItemEditor) :void
+ onChange() :void
+ onChangeFilhos(AItemEditor) :void
+ onRemoveFilho(AItemEditor) :void
AItemEditorEncaixeQuadrado
ItemEditorSpline
+ corPoliedro :THREE.Color
+ listaPontos :Array
+ poliedro :boolean
+ qtdPontos :int
+ tipoSpline :int
+ afterChangeNome(String) :void
+ onAddFilho(AItemEditor) :void
+ onChange() :void
+ onChangeFilhos(AItemEditor) :void
+ onRemoveFilho(AItemEditor) :void
AItemEditorEncaixeQuadrado
ItemEditorPoligono
+ listaPontos :Array
+ pontos :THREE.Vector3
+ pontoSelecionado :int
+ primitiva :int
+ qtdPontos :int
+ afterChangeNome(String) :void
+ getListaPontos() :Array
+ onAddFilho(AItemEditor) :void
+ onChange() :void
+ onChangeFilhos(AItemEditor) :void
+ onRemoveFilho(AItemEditor) :void


«interface»
js.cg.core::IEditorSubject
GeradorCodigoFonteJOGL
tipoGrafico : int
item : AltemEditor
UI.Panel
PainelPropriedades
- objectAngulo :UI.Number
- objectCorFundo :UI.Color
- objectCorFundoLuz :UI.Color
- objectDistancia :UI.Number
- objectEixos :UI.Checkbox
- objectExpoente :UI.Number
- objectGrade :UI.Ckeckbox
- objectIntensidade :UI.Number
- objectListaPontos :UI.Select
- objectListaPrimitiva :UI.Select
- objectListaTipoGrafico :UI.Select
- objectPonto1X :UI.Number
- objectPonto1Y :UI.Number
- objectPonto1Z :UI.Number
- objectPonto2X :UI.Number
- objectPonto2Y :UI.Number
- objectPonto2Z :UI.Number
- objectPonto3X :UI.Number
- objectPonto3Y :UI.Number
- objectPonto3Z :UI.Number
- objectPonto4X :UI.Number
- objectPonto4Y :UI.Number
- objectPonto4Z :UI.Number
- objectPontoX :UI.Number
- objectPontoY :UI.Number
- objectPontoZ :UI.Number
- objectPosicaoTargetX :UI.Number
- objectPosicaoTargetY :UI.Number
- objectPosicaoTargetZ :UI.Number
- objectQtdPontos :UI.Input
- objectTipoLuz :UI.Select
- objectTipoSpline :UI.Select
- tipoLuzAtual :int
+ criarPainelIluminacao(AltemEditor) :void
+ criarPainelTarget(AltemEditor) :void
signals : Signals
UI.Panel
Editor
+ tipoGrafico :int
editor : Editor
UI.Panel
PainelCodigoFonte
editor : Editor
UI.Panel
PainelArquivo
editor : Editor
UI.Panel
PainelListaItens
«interface»
js.cg.core::IEditorObserver
editor : Editor
signals : Signals
UI.Panel
VisualizadorGrafico
- listaBBox :Array
- listaLuzes :Array
- listaObjetosSelecionaveis :Array
- mouse :Array
- tipoGrafico :int
- addBBox(AltemEditor, THREE.Object3D) :void
- desenharEixos(THREE.Scene) :void
- desenharGrade(THREE.Scene) :void
- desenharIluminacao(AltemEditor, THREE.Object3D) :void
- desenharPoligono(AltemEditor, THREE.Object3D) :void
- desenharSpline(AltemEditor, THREE.Object3D) :void
- getBoundingBox(THREE.Object3D) :void
- selecionarObjeto(event) :void
labels : List<UI.Panel>
panels : List<UI.Panel>
ControladorPaineis
+ onChangeItemEmEdicao(AltemEditor, int) :void
0..1
1
UI.Panel
PainelStats
+ fps :Stats
+ painel :UI.Panel
+ getFrameViewer() :domElement
+ updateFPS() :void
UI.Panel
PainelMenuAjuda
UI.Panel
PainelAbrir
UI.Panel
PainelAjuda


Usuário
Editor
AObjetoGrafico
ItemEditorRenderizador
VisualizadorGrafico
onMouseUp()
Encaixou a peça
add(objeto)
onChange()
notificarOnChageItems()
onChangeItems()
visualizarItem(AItemEditor, scene)
opt item.id
[RENDERIZADOR]
visualizarItem(AItemEditor, scene)
[POLIGONO]
desenharPoligono(AItemEditor, scene)
[SPLINE]
desenharSpline(AItemEditor, scene)
[ILUMINACAO]
desenharIluminacao(AItemEditor, scene)

Desenvolvimento
Ferramentas

- **Eclipse** – Java EE IDE for Web Developers
- **Google Chrome** – 35.0.1916.114m
- **Mozila Firefox** – 30.0
- **Opera** – 22.0
- **Enterprise Architect**
- **BitBucket**
- **SourceTree**

```
...
//POLIGONO
else if (item.primitiva == CG.listaDePrimitivas.Preenchido) {
      // DESENHA A LINHA ANTES DAS FACES
      var geoLine = new THREE.Geometry();
      geoLine.vertices = points;
      geoLine.vertices.push(item.listaPontos[0]); //REPETE O PRIMEIRO PONTO

      var matLine = new THREE.LineBasicMaterial({linewidth: 2,
                                                 color: cor,
                                                 transparent: false});
      var line = new THREE.Line(geoLine, matLine, THREE.LineStrip);
      line.item = item;
      objetoAux.add(line);
      scope.listaObjetosSelecionaveis.push(line);

      var geometria = new THREE.Geometry();
      geometria.vertices = points;

      //ADICIONA AS FACES DOS TRIANGULOS
      for (var i = 0; i < (item.listaPontos.length - 2); i++) {
            geometria.faces.push(new THREE.Face3(0, i + 1, i + 2));
      }

      geometria.computeLineDistances();

      var material = new THREE.MeshPhongMaterial({color:cor,
                                                  ambient: cor,
                                                  overdraw: true,
                                                  side:THREE.DoubleSide});

      var poligono = new THREE.Mesh(geometria, material);

      poligono.item = item;
      objetoAux.add(poligono);
      scope.listaObjetosSelecionaveis.push(poligono);
}
...
```

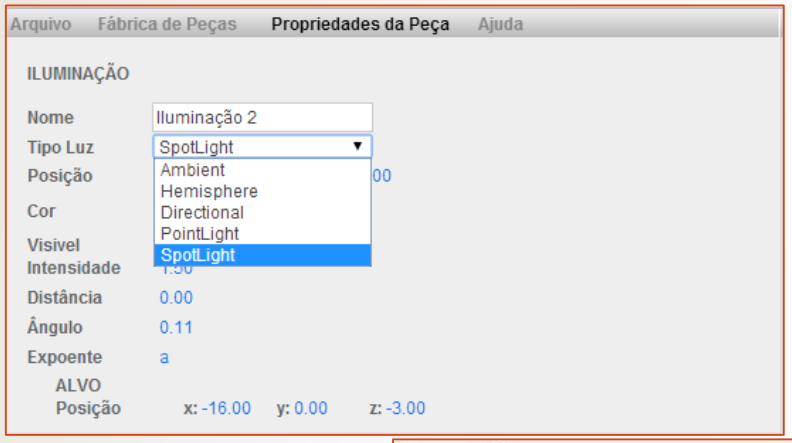
```
...
//SPLINE
var pontosSpline = [];

var t;
var p0, p1, p2, p3;
var x, y, z;

for (var i = 0; i <= item.qtdPontos; i++) {
      t  = (i / item.qtdPontos);
      p0 = Math.pow((1 - t), 3);
      p1 = (3 * t * Math.pow((1 - t), 2));
      p2 = (3 * Math.pow(t, 2) * (1 - t));
      p3 = Math.pow(t, 3);

      x = (p0 * points[0].x + p1 * points[1].x + p2 * points[2].x + p3 * points[3].x);
      y = (p0 * points[0].y + p1 * points[1].y + p2 * points[2].y + p3 * points[3].y);
      z = (p0 * points[0].z + p1 * points[1].z + p2 * points[2].z + p3 * points[3].z);

      pontosSpline.push(new THREE.Vector3(x, y, z));
}

var geometria      = new THREE.Geometry();
geometria.vertices = pontosSpline;
geometria.computeLineDistances();

var material = new THREE.LineBasicMaterial({linewidth: 2,
                                            color: cor,
                                            transparent: false });
var spline = new THREE.Line(geometria, material, THREE.LineStrip);

objetoAux.add(spline);
spline.item = item;
scope.listaObjetosSelecionaveis.push(spline);

spline.add(addBBox(item, spline));
...
```

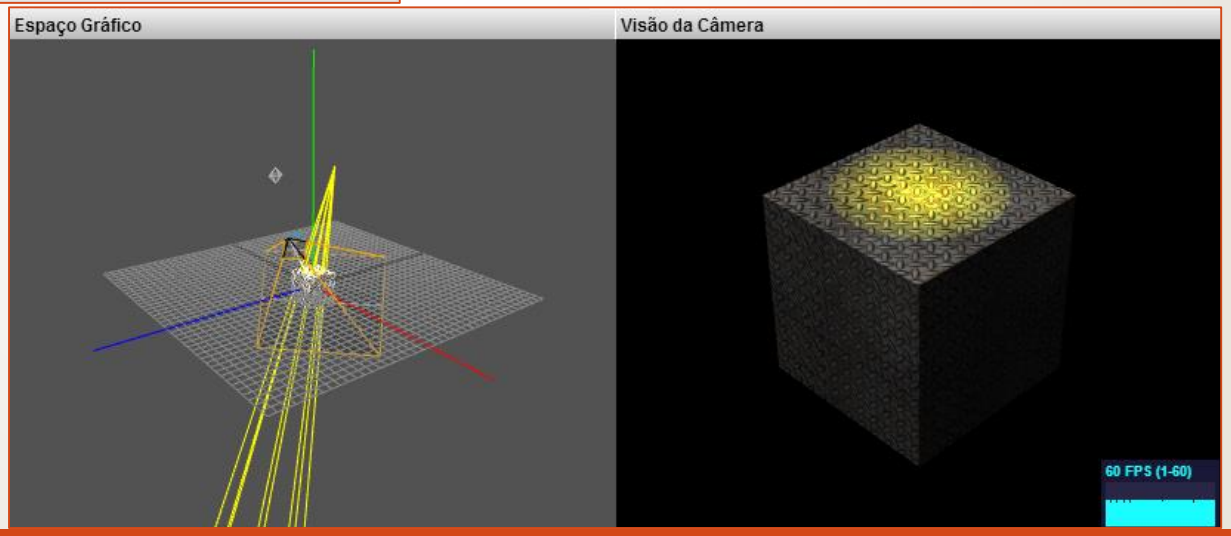
```
...
//ILUMINACAO
else if (item.tipoLuz == CG.listaTipoLuz.SpotLight) {
      light = new THREE.SpotLight(cor, item.intensidade,
                                        item.distancia,
                                        item.angulo,
                                        item.expoente);

      light.position.set(item.posicao.x, item.posicao.y, item.posicao.z);

      light.target.position.set(item.posicaoTarget.x,
                                item.posicaoTarget.y,
                                item.posicaoTarget.z);

      light.target.matrixWorld.elements[12] = item.posicaoTarget.x;
      light.target.matrixWorld.elements[13] = item.posicaoTarget.y;
      light.target.matrixWorld.elements[14] = item.posicaoTarget.z;

      helper = new THREE.SpotLightHelper(light, 10);
}

if (light !== undefined) {
      objetoAux.add(light);
      listaLuzes.push(light);
}

if (helper !== undefined) {
      sceneHelper.add(helper);
      helper.update();
      listaLuzesHelpers.push(helper);
}
...
```

Desenvolvimento
Seleção no Espaço Gráfico

```
1  function selecionarObjeto(event) {
2    mouse.x = ((event.clientX - scope.dom.offsetTop) / (renderer.domElement.width)) * 2 - 1;
3    mouse.y = -((event.clientY - scope.dom.offsetTop) / renderer.domElement.height) * 2 + 1;
4
5    var vetor    = new THREE.Vector3(mouse.x, mouse.y, 0);
6    var projetor = new THREE.Projector();
7    projetor.unprojectVector(vetor, views[0].camera);
8
9    var raycaster = new THREE.Raycaster();
10   raycaster.set(views[0].camera.position, vetor.sub(views[0].camera.position).normalize());
11
12   var intersects = raycaster.intersectObjects(scope.listaObjetosSelecionaveis);
13
14   if (intersects.length > 0) {
15     editor.selecionarItem(intersects[0].object.item);
16   }
17   else {
18     editor.selecionarItem(null);
19
20     scope.listaObjetosSelecionaveis = [];
21     visualizarItem(scope.editor.painelMontagem, scene);
22   }
23 };
```

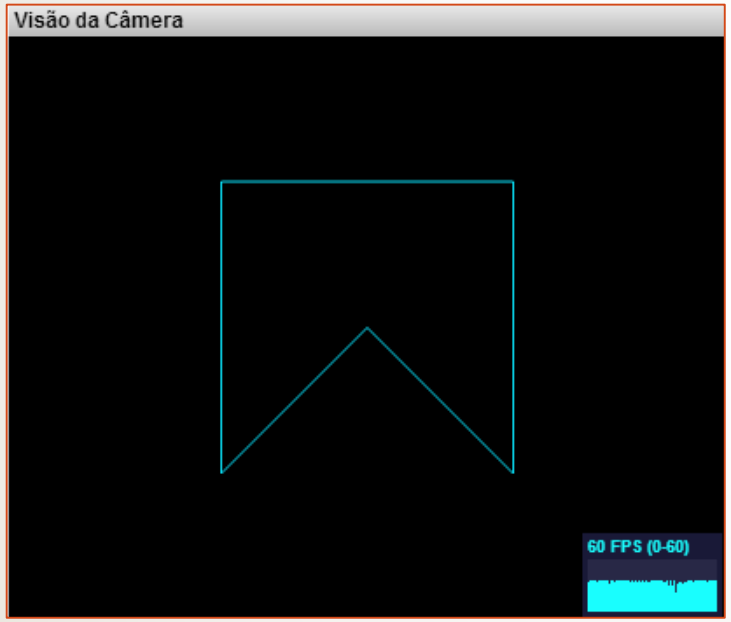
Desenvolvimento
Operacionalidade da Implementação



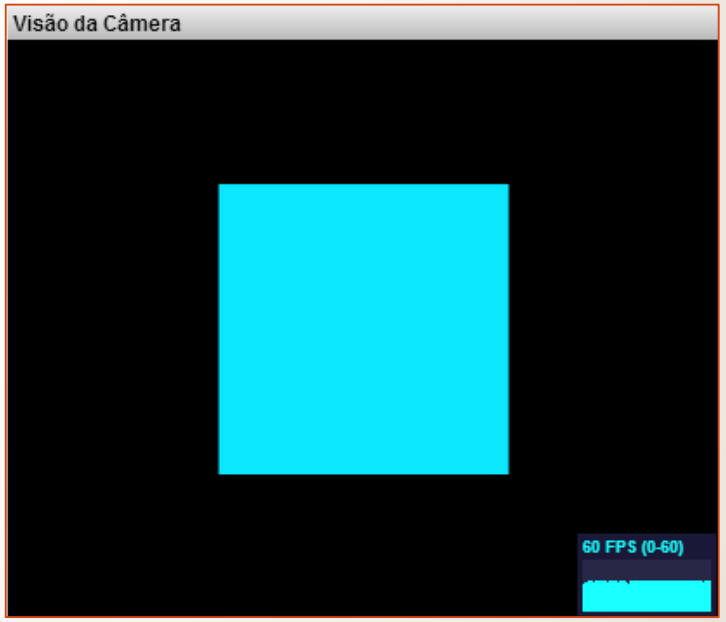


Resultados e Discussões
Limitações

- Polígonos côncavos



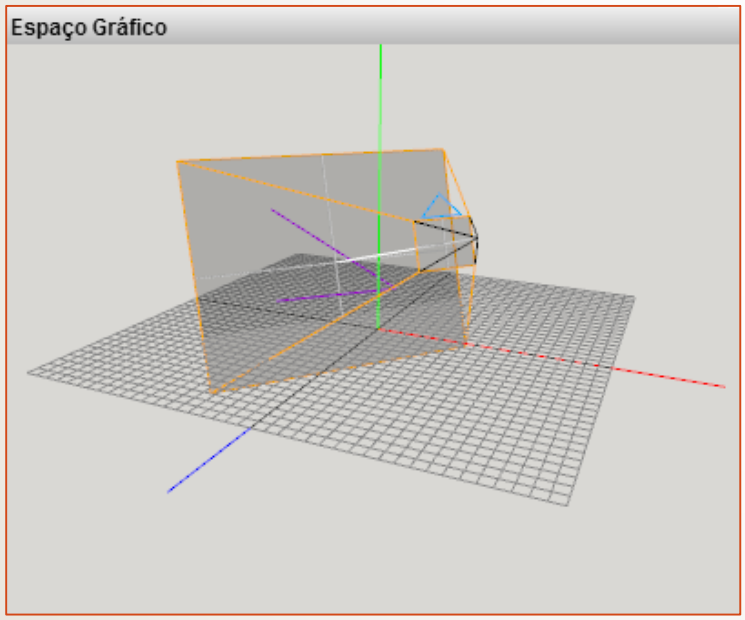
Fechado

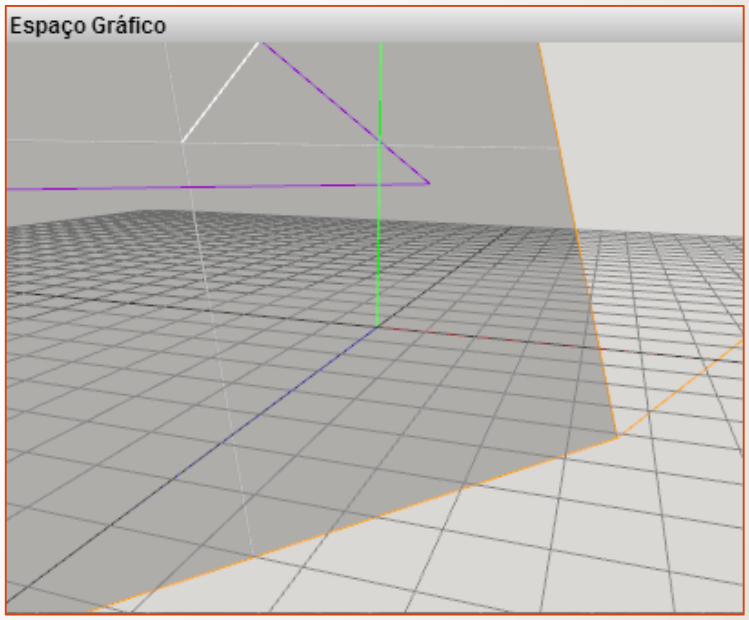


Preenchido

Resultados e Discussões
Limitações

- Seleção de linhas



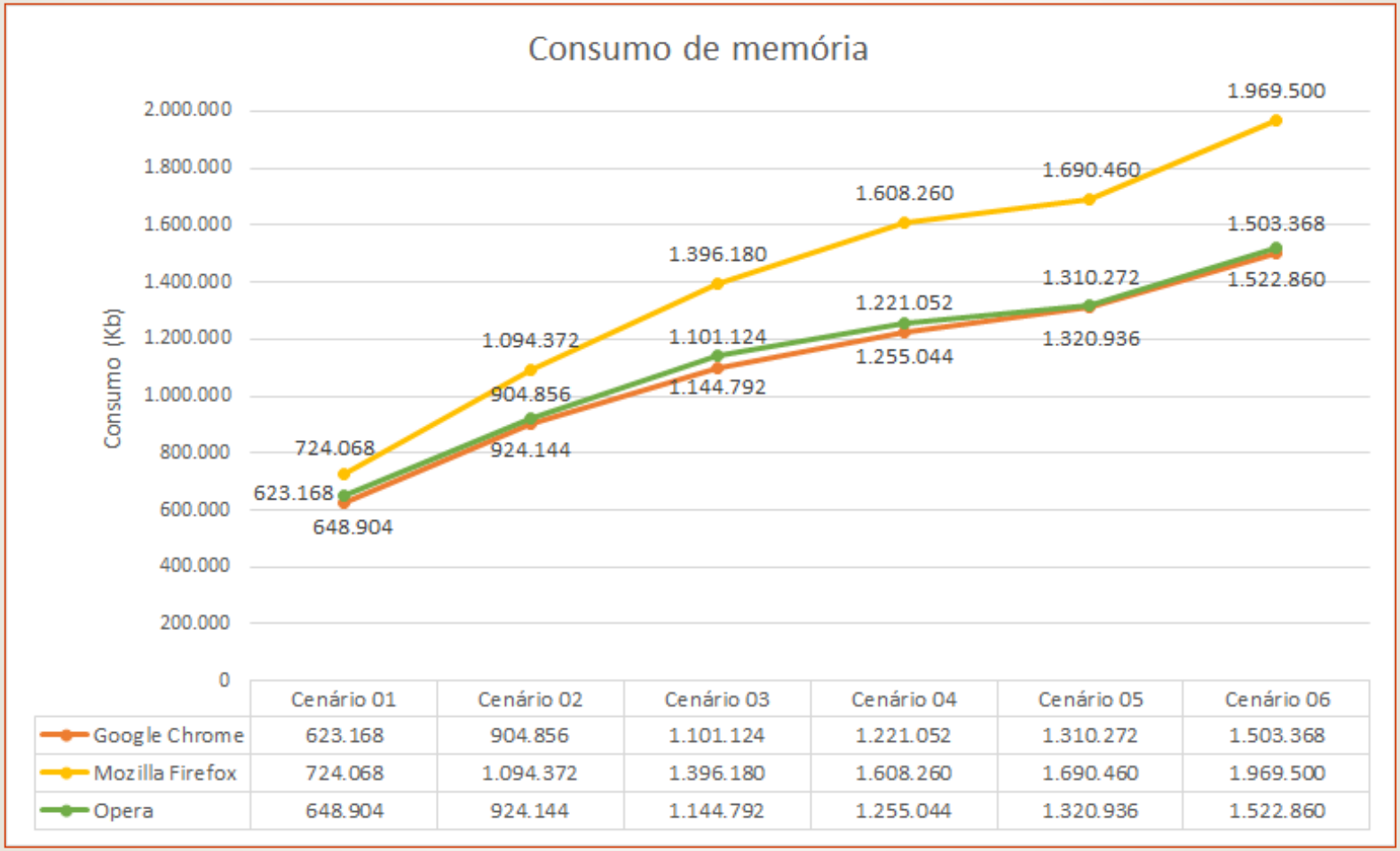


Resultados e Discussões
Teste de Desempenho

- Cenários
- Processamento de FPS
- Consumo de memória

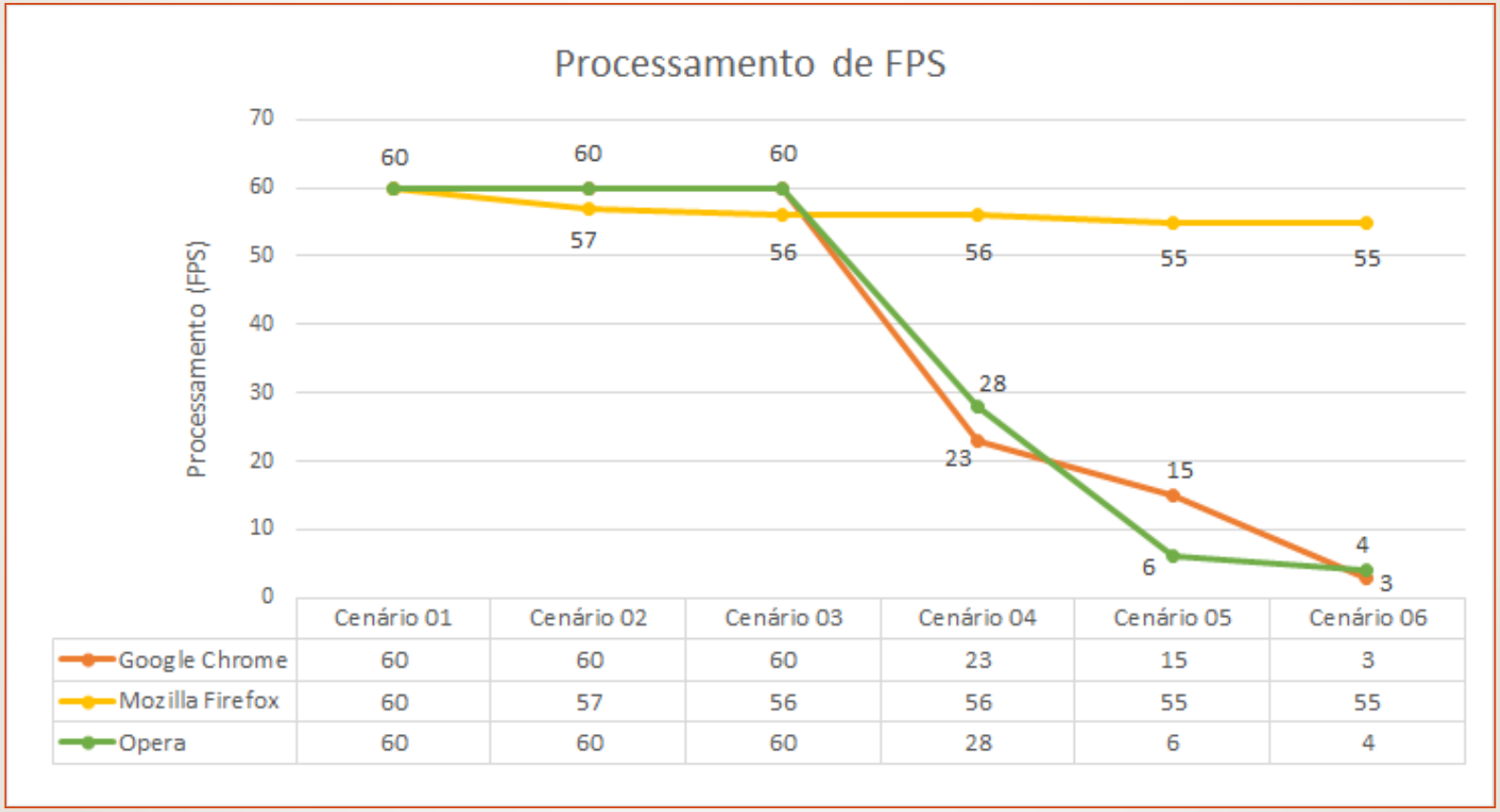
Peça	Cenário 01	Cenário 02	Cenário 03	Cenário 04	Cenário 05	Cenário 06
Renderizador	1	1	1	1	1	1
Câmera	1	1	1	1	1	1
Iluminação	1	1	1	1	1	2
Objeto Gráfico	3	6	9	11	12	15
Transladar	0	6	9	11	12	15
Cubo	1	2	3	3	4	5
Polígono	1	2	3	4	4	5
Spline	1	2	3	4	4	5
Total	9	21	30	36	39	49

Resultados e Discussões



	Cenário 01	Cenário 02	Cenário 03	Cenário 04	Cenário 05	Cenário 06
Google Chrome	623.168	904.856	1.101.124	1.221.052	1.310.272	1.503.368
Mozilla Firefox	724.068	1.094.372	1.396.180	1.608.260	1.690.460	1.969.500
Opera	648.904	924.144	1.144.792	1.255.044	1.320.936	1.522.860

Resultados e Discussões



	Cenário 01	Cenário 02	Cenário 03	Cenário 04	Cenário 05	Cenário 06
Google Chrome	60	60	60	23	15	3
Mozilla Firefox	60	57	56	56	55	55
Opera	60	60	60	28	6	4

Resultados e Discussões

- **Testes práticos com a turma de CG**
  - Realização de uma atividade na ferramenta
  - Questionário após realização da atividade
  - Atividade aplicada numa turma de 27 alunos, sendo que 24 alunos fizeram a atividade e 18 responderam o questionário

Notas	0,0	0,1	0,2	0,3	0,4	0,5	0,6	0,7	0,8	0,9	1,0	Total
Alunos	2	0	1	0	0	0	4	1	5	1	10	24

Resultados e Discussões

O que você achou da "operacionalidade" do aplicativo?	
Difícil, nada intuitivo	0%
Mediano, mas pouco intuitivo	28%
Mediano mas intuitivo	39%
Fácil e intuitivo	33%

O aplicativo ajudou a entender a aplicação de como usar os objetos gráficos "Cubo", "Polígono" e "Spline"?	
Não ajudou	0%
Ajudou pouco	22%
Ajudou	50%
Ajudou muito	28%

Resultados e Discussões

O aplicativo ajudou a entender a aplicação de "iluminação" nos objetos gráficos?	
Não ajudou	0%
Ajudou pouco	6%
Ajudou	61%
Ajudou muito	33%

O aplicativo possibilitou entender em 2D os mesmos conceitos gráficos apresentados no espaço 3D?	
Não ajudou	0%
Ajudou pouco	0%
Ajudou	78%
Ajudou muito	22%

Resultados e Discussões
Trabalhos correlatos

Característica	StarLogo TNG	Motor de jogos 3D	VisEdu-CG 2.0	VisEdu-CG 3.0
Diversos tipos de objetos gráficos	X	X	-	X
Iluminação de cena	-	X	-	X
Animação da cena	X	-	-	-
Seleção de objetos no espaço gráfico	X	-	-	X
Exportação/Importação em formato de arquivo conhecido	X	-	X	X
Disponibilidade na web	-	X	X	X
Janelas e painéis dinâmicos	X	-	-	-
Tipo de cenário	3D	3D	3D	2D/3D

Conclusões

- Objetivos atendidos
  - Novos objetos gráficos Polígono e Spline
  - Uso da Iluminação na representação gráfica
  - Seleção dos objetos no Espaço Gráfico
- Avaliação positiva dos usuários

Extensões

- Utilização de VBOs
- Painéis dinâmicos
- Animação da cena
- Espaço gráfico normalizado
- Ajustar do polígono Preenchido
- Faces do polígono pré-calculadas
- Pontos da Spline pré-calculados

Apresentação Prática

FIM.
Obrigado!

Samuel Anderson Nunes
samuel.anderson.nunes@gmail.com