Roteiro

- **Introdução**
- **Objetivo do trabalho**
- **Fundamentação teórica**
  - **Controle de tráfego viário**
  - **Software Simulador do controle de tráfego de automóveis em uma malha rodoviária urbana**
  - **Editores gráficos**
  - **Computação gráfica**
  - **Biblioteca gráfica OpenGl**
  - **Geometria Analítica e Trigonometria**

Roteiro

- **Desenvolvimento**
  - **Requisitos do Sistema**
  - **Especificação do Sistema**
  - **Implementação**
  - **Operacionalidade do protótipo**
  - **Resultados e Discussão**
  - **Conclusão**
  - **Extensões**

Introdução

- No primeiro semestre de 2004 foi desenvolvido um sistema que simula o tráfego de veículos em uma área urbana por Jocemar J. Freire (FREIRE, 2004).
- Dificuldade em especificar a malha rodoviária

Objetivos do trabalho

O objetivo deste trabalho é a construção de um editor gráfico 2D que crie uma malha viária graficamente, e após, armazená-la em um arquivo texto compatível para o sistema de controle de trânsito desenvolvido por Freire.

CONTROLE DE TRÁFEGO VIÁRIO

Segundo Associação brasileira de monitoramento e controle eletrônico de trânsito (2000), pode-se definir Controle de Tráfego Viário como a supervisão do movimento de veículos e pessoas, com o objetivo de garantir eficiência e segurança. Uma rodovia pode ser considerada eficiente quando fornece aos seus usuários a possibilidade de se movimentar ao menor custo possível. É vista como segura ao reduzir ou eliminar os acidentes na sua totalidade.

SOFTWARE SIMULADOR DO CONTROLE DE TRÁFEGO DE AUTOMÓVEIS EM UMA MALHA ROVIÁRIA URBANA

O Trabalho desenvolvido por Freire em 2004, propõe simular e identificar em uma determinada malha rodoviária situações de engarrafamento

Editores Gráficos

- Técnicas para construção de interface de usuário são aplicadas exaustivamente em software do tipo editores de textos ou gráficos.
- Os editores gráficos adotam como características fundamentais:
- ser consistente;
- fornecer feedback;
- minimizar erros;
- delimitar área de desenho;

Computação Gráfica

- "A Computação Gráfica é parte da Ciência da Computação e área de estudo de alguns aspectos da comunicação entre o homem e o computador. O aspecto principal abordado pela Computação Gráfica é o da comunicação visual no sentido máquina-homem, através da síntese de imagens em dispositivos de saída apropriados" (BANON, 1989, p. 1).

Biblioteca Gráfica OpenGl

- OpenGL é uma biblioteca gráfica padrão de alto desempenho, independente do sistema de janelas e do hardware gráfico utilizado, voltada para manipulação e renderização de objetos tridimensionais;

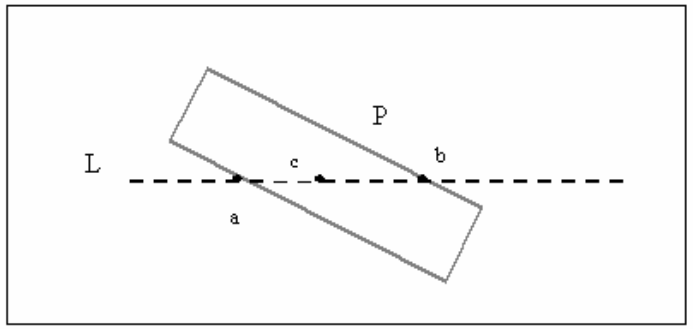
- A biblioteca funciona como uma máquina de estados. Todos os estados ou modos habilitados nas aplicações têm efeito enquanto os mesmos estiverem ligados ou forem modificados.

Geometria Analítica e Trigonometria

- A **geometria analítica** é um ramo da matemática que estuda figuras geométricas, analisando-as através de elementos e processos algébricos.
- **Trigonometria** é o ramo da matemática que trata das relações entre os lados e ângulos de triângulos (polígonos com três lados).
- Funções para cálculo da **distância entre dois pontos**, o **ponto médio de um segmento, coeficiente angular, Interseção de duas Retas e Ponto em Polígono.**

Ponto em Polígono

```
Fronteira[1]:= PtA;   Fronteira[2]:= PtB;   Fronteira[3]:= PtC;
Fronteira[4]:= PtD;   Fronteira[5]:= PtA;
n:=0;
// teste cada lado do polígono
For i:=1 to 4 do
begin
   PtE:= Fronteira[i];
   PtF:= Fronteira[i+1];
   // lado não é horizontal
   If (abs(PtE.Y - PtF.Y) > Erro) and (abs(PtF.X - PtE.X) > Erro) then
    begin
       Xinter:= (PtY - PtE.Y + (PtF.Y - PtE.Y)/(PtF.X-PtE.X)*PtE.X)/
                  ((PtF.Y - PtE.Y)/(PtF.X - PtE.X));
   //PtE é da Fronteira... PARE
   If (abs(PtX - PtE.X)) < Erro then
   begin
      n:= 1;
      break;
   end
```



Ponto em Polígono

```
// Não têm y mínimo
else
        If (Xinter > PtX) and (PtY > min(PtE.Y,PtF.Y)) and (PtY <= max(PtE.Y,PtF.Y)) then
          n:= n + 1;
      end
// ponto é da fronteria horizontal... PARE
else
  If (abs(PtE.Y - PtF.Y) > Erro) then
         PtG:= Fronteira[3];
         // VERTICAL
         if (PtY >= min(PtE.Y,PtG.Y)) and (PtY <= max(PtE.Y,PtG.Y)) and
            (PtX >= min(PtE.X,PtF.X)) and (PtX <= max(PtE.X,PtF.X)) then
         begin
           n:= 1;
          break;
         end
        else
        if (abs(PtF.X - PtE.X) > Erro) then
          PtG:= Fronteira[3];
```

Ponto em Polígono

```
//HORIZONTAL
        if  (PtY >= min(PtE.Y,PtF.Y)) and (PtY <= max(PtE.Y,PtF.Y)) and
            (PtX >= min(PtE.X,PtG.X)) and (PtX <= max(PtE.X,PtG.X)) then
        begin
            n:= 1;
            break;
        end;
 end;
//end For
if ((n mod 2) <> 0) then
        break;
    end; // end For Malha
 end;
// Verdadeiro se ímpar
  impar:= ((n mod 2) <> 0);
If impar then
   Result:=z
  else
   Result:=99;
end;
```

Requisitos do Sistema

- permitir a modelagem da malha rodoviária (RF);
- permitir a criação, modificação e exclusão de um trecho (RF);
- permitir a junção de dois trechos (RF);
- permitir a divisão de um trecho (RF);
- permitir fazer trechos paralelos (RF);

Requisitos do Sistema

- permitir a opção de mudança de tamanho da malha (RF);

- permitir a opção de mover mapa no mundo (RF);

- apresentar as informações de forma perceptível (facilidade de identificação dos objetos) pelo usuário do sistema (RNF).

Diagrama de Atividades

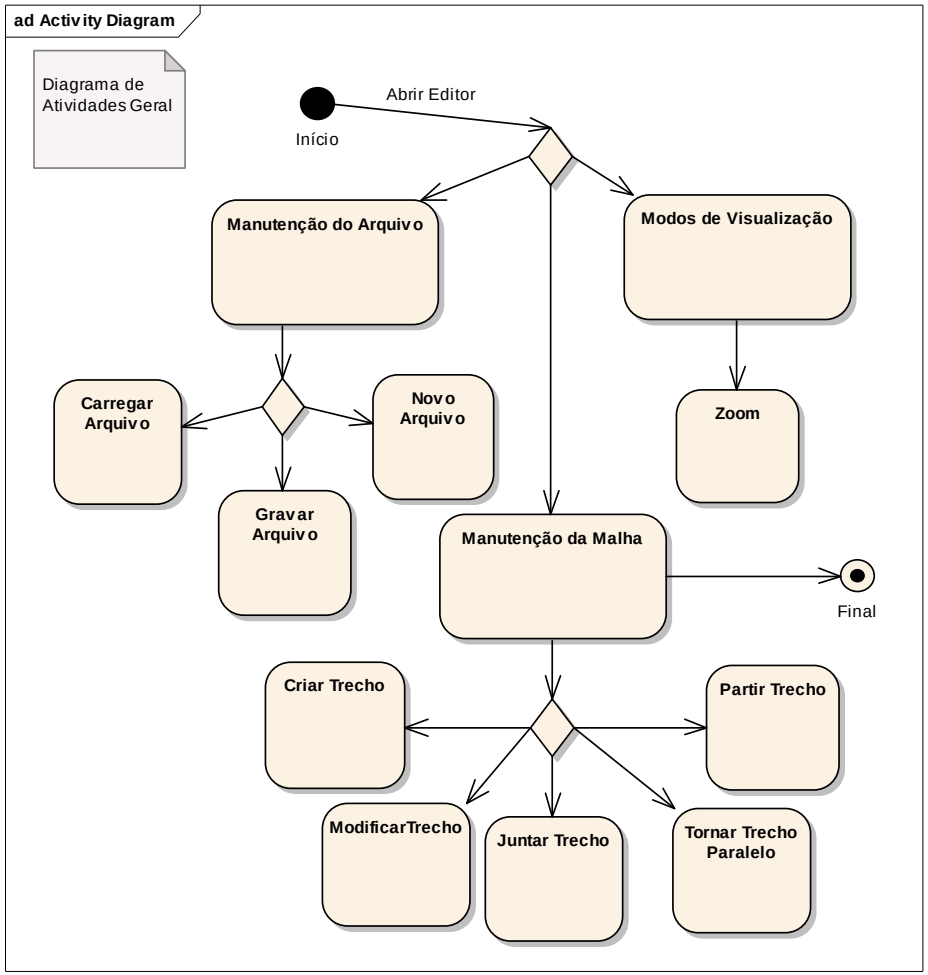


Diagrama de Estado "Criar Trecho"

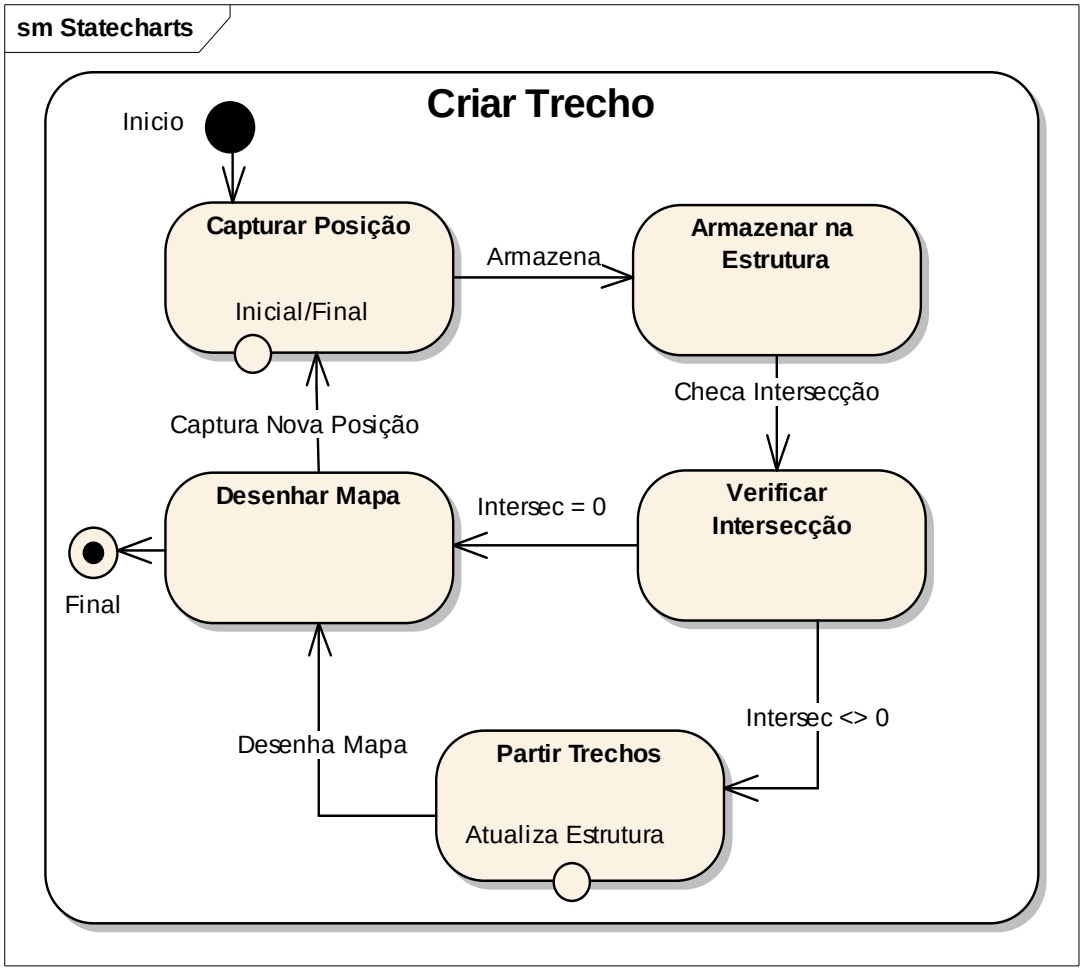


Diagrama de Estado "Modificar Atributos"

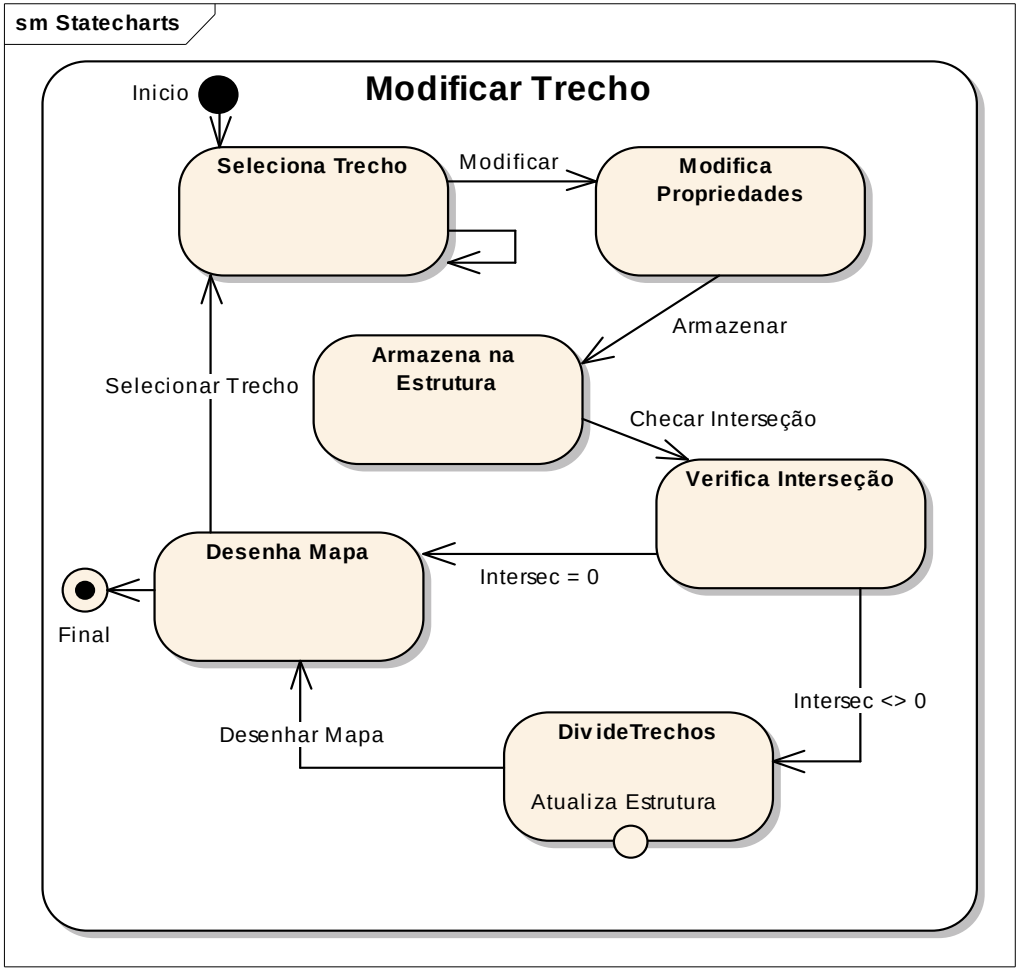


Diagrama de Estado "Partir Trecho"

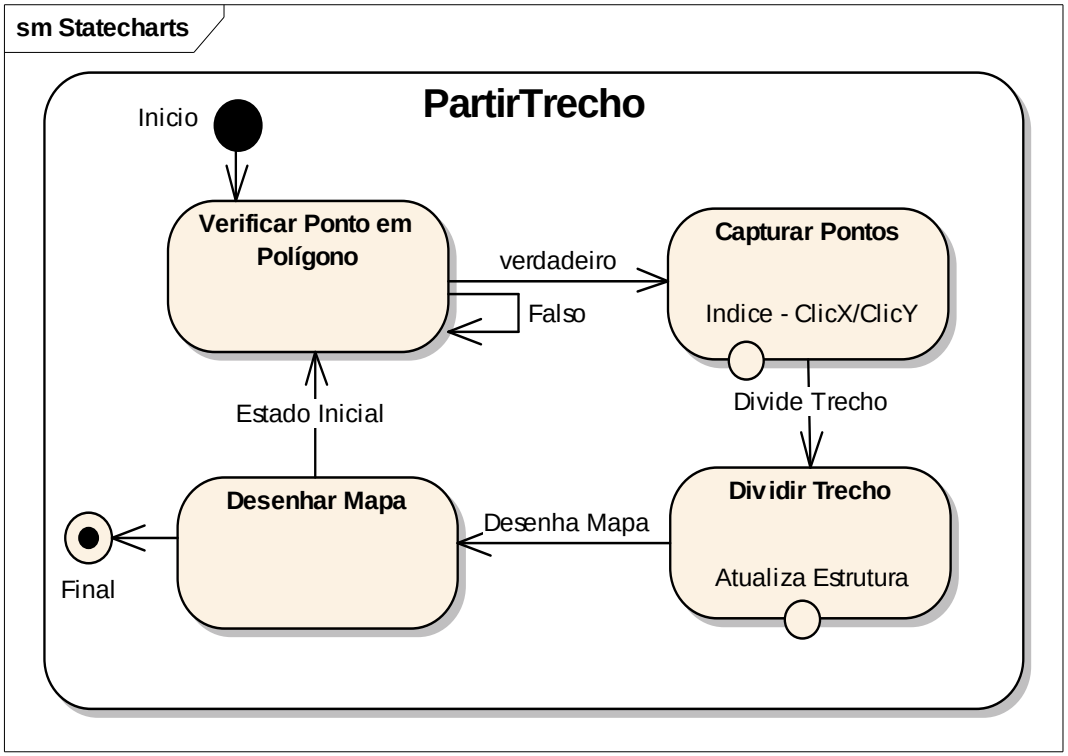


Diagrama de Estado "Juntar Trecho"

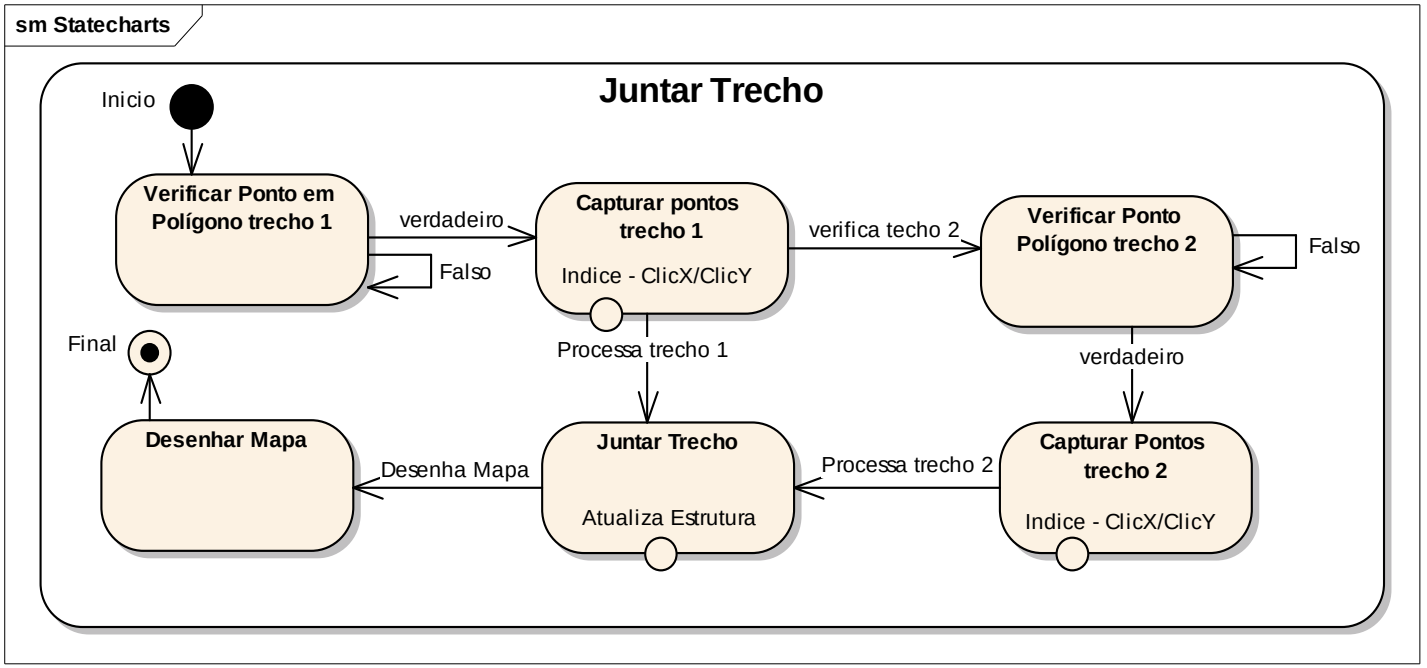


Diagrama de Estado "Tornar Trecho Paralelo"

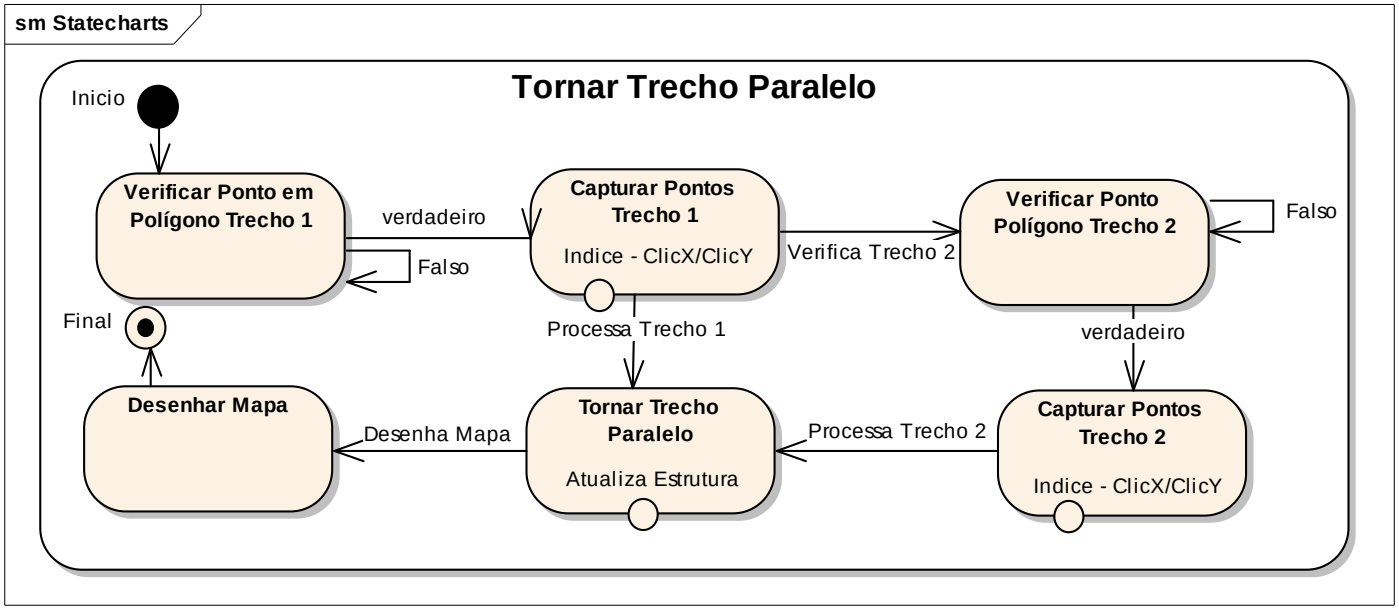
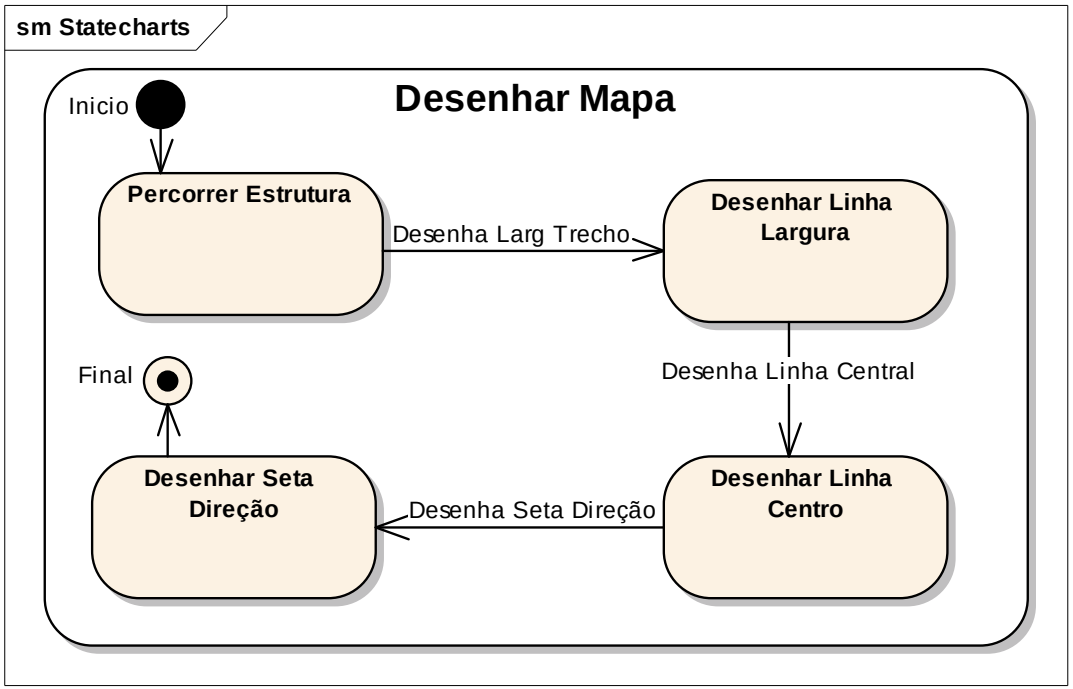


Diagrama de Estado "Desenhar Mapa"

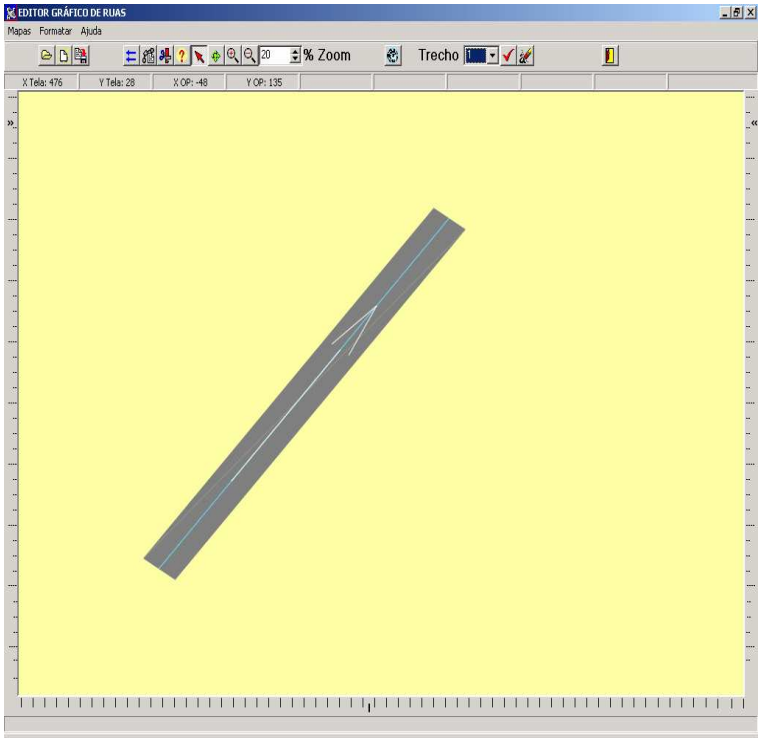


Implementação

- O protótipo foi implementado na ferramenta de programação Delphi, usando a biblioteca gráfica OpenGl.

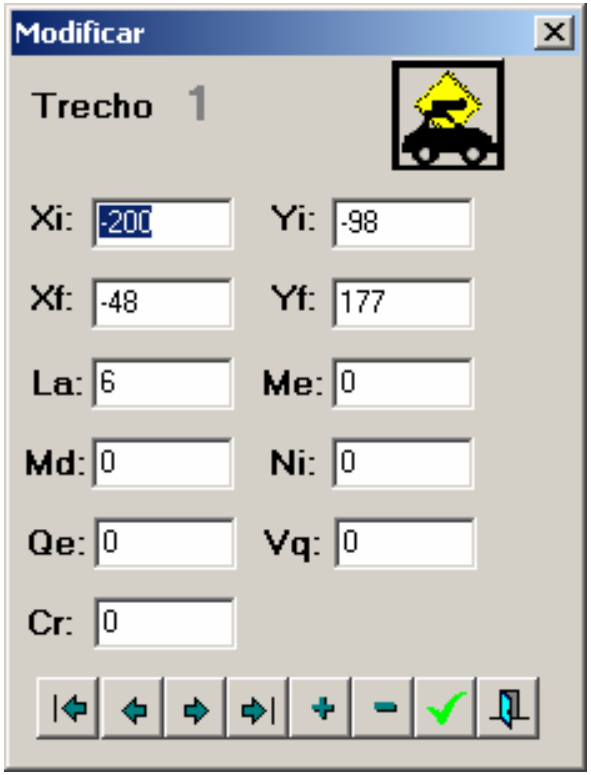
Rotina "Criar Trecho"

```
// ROTINA CRIAR  TRECHO
// Se botão da reta estiver pressionado.
if FrmPrincipal.BtnReta.Down then
  begin
    // Guarda ponto final do click
    ClickF.X:= PtX;
    ClickF.Y:= PtY;
    // Armazena na estrutura fila.
    Guarda_Retas(indice,Click,ClickF,6,false);
    // Verifica se existe interseção.
    calcIntersec;
    // Incrementa contador
    indice:= L_Malha.Count+1;
    // Atualiza ComboBox
    frmCheck.AtualizarComboBox;
   // desabilita desenho da reta
    reta:= false;
    // redesenha mapa
    redesenha;
   end;
   // FIM DA ROTINA
```



Rotina "Modificar Atributos"

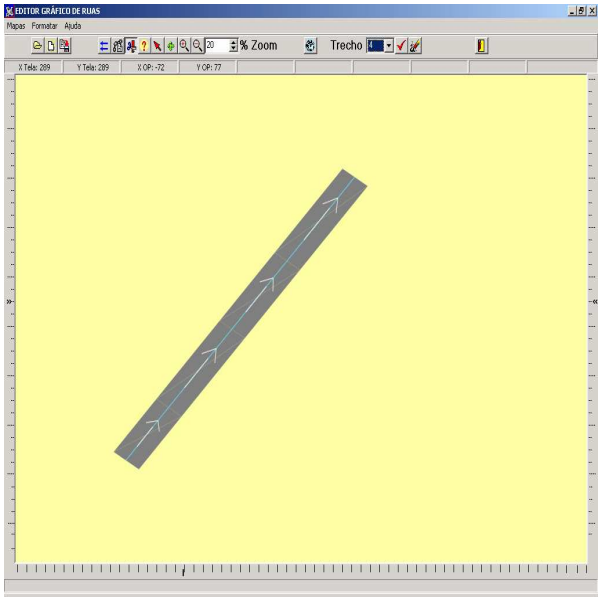
```
if inserir then    //ROTINA MODIFICAR ATRIBUTOS DO TRECHO
  begin
    // Armazena informações da Janela na Lista
    Guarda_RetasEx(indice,StrToInt(edtmd.Text),
StrToInt(edtme.Text),StrToInt(edtni.Text),
    StrToInt(edtqe.Text), StrToInt(edtvq.Text), StrToFloat(edtxi.Text),
StrToFloat(edtyi.Text),StrToFloat(edtxf.Text),StrToFloat(edtyf.Text),
StrToFloat(edtlargura.Text),false);
    AtualizarComboBox;
    indice:=L_Malha.Count+1;
    inserir:=False;
  end else
  begin
    Ponteiro.I_CdTrecho:=StrToInt(edtReta.Text);
    Ponteiro.Inicio.X:=StrToFloat(edtXi.Text);
    Ponteiro.Inicio.Y:=StrToFloat(edtYi.Text);
    Ponteiro.Fim.X:=StrToFloat(edtXf.Text);
    Ponteiro.Fim.Y:=StrToFloat(edtYf.Text);
    Ponteiro.MDir:=StrToInt(edtmd.Text);
    Ponteiro.MEsq:=StrToInt(edtme.Text);
    Ponteiro.Nivel:=StrToInt(edtni.Text);
    Ponteiro.Quebra:=StrToInt(edtqe.Text);
    Ponteiro.Vl_Quebra:=StrToInt(edtvq.Text);
    Ponteiro.F_Largura:=StrToFloat(edtlargura.Text);
    Ponteiro.Cruza:= StrToBool(edtcruza.Text);
  end;
  // Redesenha conteúdo da tela.
  FrmGrafico.Redesenha;
                    // FIM DA ROTINA
```



Desenvolvimento

Rotina "Partir Trecho"

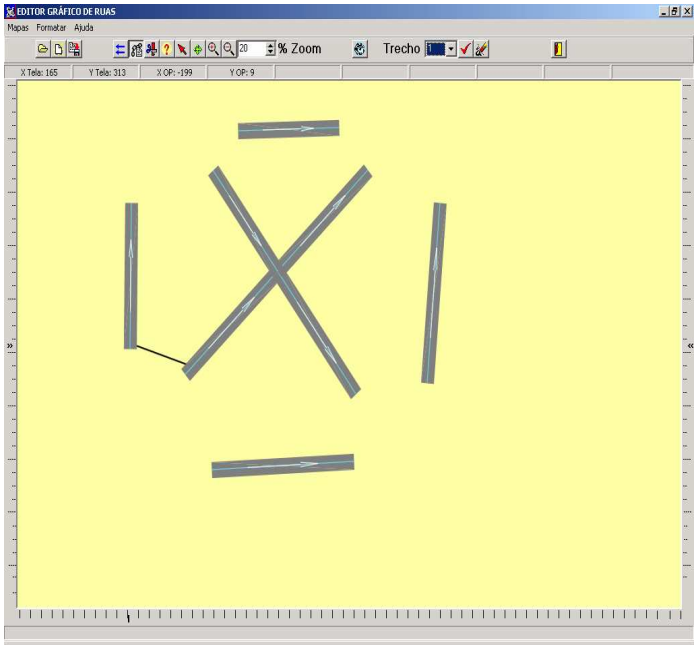
```
Begin                                  // ROTINA PARTIR TRECHO
   If L_Malha <> nil then
   begin
      For z:= 1 to L_Malha.Count do
      begin
         If z = indice then
         begin
           // Percorre Lista
            Ponteiroz:= L_Malha.Items[z-1];
           //  Calcula Coeficiente Angular
           CAng:= (Ponteiroz.Fim.Y-Ponteiroz.Inicio.Y)/Aux.
           // Calcula Coeficiente Linear
        CLin:= Ponteiroz.Inicio.Y-(CAng * Ponteiroz.Inicio.X);
           Aux:= Ponteiroz.Fim.X-Ponteiroz.Inicio.X;
          // Se o trecho é vertical, Xi = Xf , captura apenas Y para corte.
          If Aux = 0 then
          begin
             AuxFimX:= Ponteiroz.Fim.X;
             AuxFimY:= Ponteiroz.Fim.Y;
             Ponteiroz.Fim.Y:= PtY;
       Guarda_RetasEX(L_Malha.Count+1,0,0,0,0,0,0,AuxFimX,AuxFimY,AuxFimX,PtY,6,true);
            break;
          end;
      NovoPtY:= (CAng * PtX) + CLin; // Calcula novo Ponto Y
   // Guarda Pontos Finais
        AuxFimX:= Ponteiroz.Fim.X;
        AuxFimY:= Ponteiroz.Fim.Y;
    // Entrada de novos valores finais
        Ponteiroz.Fim.X:= PtX;
        Ponteiroz.Fim.Y:= NovoPtY;
// Atualiza Estrutura
        Guarda_RetasEX(L_Malha.Count+1,0,0,0,0,0,0,PtX,NovoPtY,AuxFimX,AuxFimY,6,true);
         end;    end;     end;    end;        //   FIM DA ROTINA
```



Desenvolvimento

Rotina "Juntar Trecho"

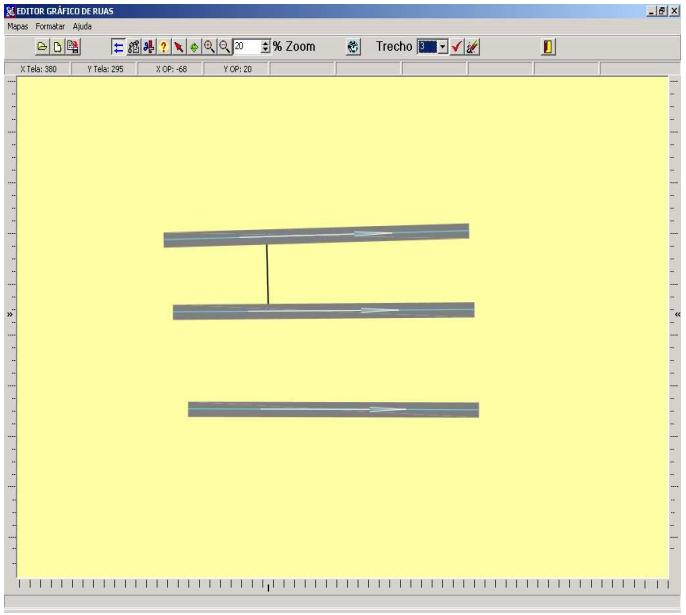
```
Begin              //ROTINA JUNTAR TRECHO
  If L_Malha <> nil then
  Begin
      // Verifica em qual das metades do trecho se encontra o ponto
cliclado e armazena posição.
      If PontoEmPoligono(Pt1.X,Pt1.Y,2)<> 99 then
        fig1:= 'inicio'
       else
        fig1:= 'fim';
      If PontoEmPoligono(Pt2.X,Pt2.Y,2)<> 99 then
        fig2:= 'inicio'
       else
        fig2:= 'fim';
      // Percorre Lista e junta partes se condições forem atendidas
      Ponteiroz:= L_Malha.Items[ind2];
      If (fig1 = 'inicio') and (fig2 = 'inicio') then
      aux:= Ponteiroz.Inicio
       else If (fig1 = 'inicio') and (fig2 = 'fim') then
            aux:= Ponteiroz.Fim;
      Ponteiroz:= L_Malha.Items[ind2];
      If (fig1 = 'fim') and (fig2 = 'inicio') then
      aux:= Ponteiroz.Inicio
       else If (fig1 = 'fim') and (fig2 = 'fim') then
            aux:= Ponteiroz.Fim;
      Ponteiroz:= L_Malha.Items[ind1];
      If (fig1 = 'inicio') then
        Ponteiroz.Inicio:= aux
      else
        Ponteiroz.fim:= aux;
  end;
end;                    //FIM DA ROTINA
```



Desenvolvimento

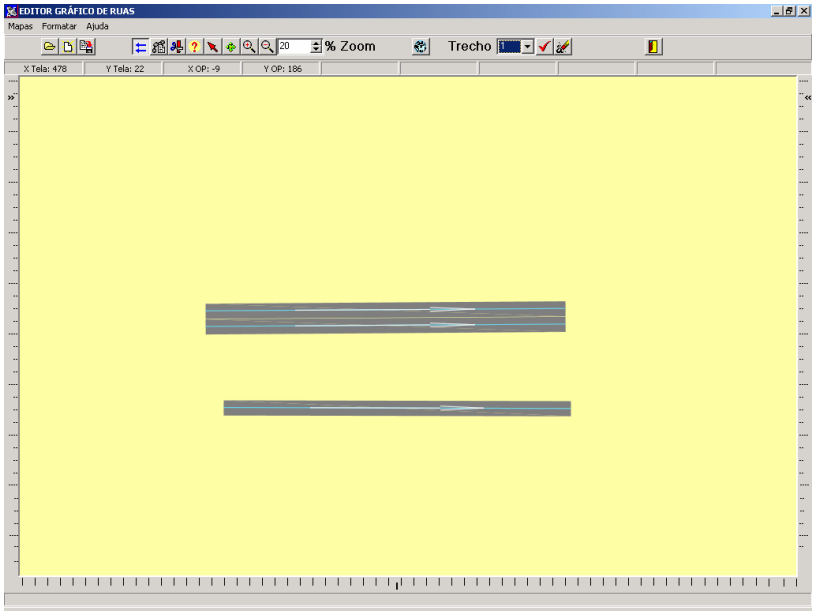
Rotina "TORNAR TRECHO PARALELO"

```
Begin                        // ROTINA TORNAR TRECHO PARALELO
  If L_Malha <> nil then
  begin
     // Percorre Lista
     ponteirofig1:= L_Malha.Items[ind1];
      ponteirofig2:= L_Malha.Items[ind2];

     // Calcula Pontos Extremos
     CalcPt(Ponteirofig2.Inicio,Ponteirofig2.Fim,Ponteirofig2.F_Largura, PtA,
PtB, PtC, PtD);
    // Calcula DX e DY
     dX:=  PtC.X - PtB.X;
     dY:=  PtC.Y - PtB.Y;
     dX1:= PtB.X - PtC.X;
     dY1:= PtB.Y - PtC.Y;
    // Calcula largura total
    larguratotal:= (ponteirofig1.F_Largura/2) + (ponteirofig2.F_Largura/2);
   // Calcula Delta X e Delta Y
    if ((ponteirofig2.Inicio.X - ponteirofig2.Fim.X) > Erro) then
    begin
      deltaX:= larguraTotal;
      deltaY:= 0;
    end else
```



Rotina "TORNAR TRECHO PARALELO"

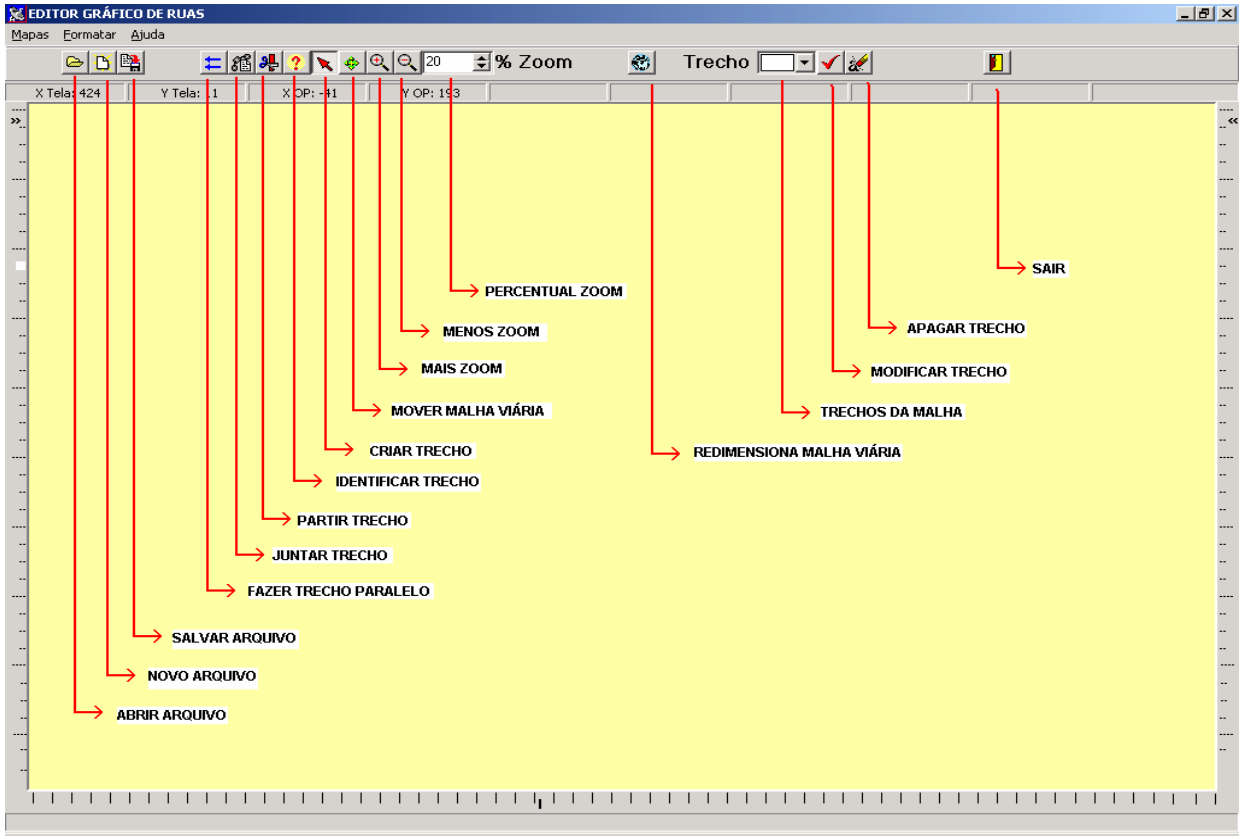
```
if ((PtB.Y - PtC.Y) < Erro) then  // CONTINUAÇÃO .....

    begin
      deltaY:= larguraTotal;
      deltaX:= 0;
    end else
          If ((dX - dY) > Erro) then
           begin
            angulo:= ArcTan(dY/dX);
            deltaX:= larguraTotal * sin (angulo);
            deltaY:= larguraTotal * cos (angulo);
           end;
  end;
  //  Armazena em variável auxiliar
  auxFimX := PtC.X     + deltaX;
  auxFimY := PtC.Y     - deltaY;
  auxIniX:=  PtB.X  + deltaX;
  auxIniY:=  PtB.Y  - deltaY;

  // Atualiza Estrutura
  ponteiroz:= L_Malha.Items[ind1];
  ponteiroz.Inicio.X:= auxIniX;
  ponteiroz.Inicio.Y:= auxIniY;
  ponteiroz.Fim.X:= auxFimX;
  ponteiroz.Fim.Y:= auxFimY

end;          // FIM DA ROTINA
```



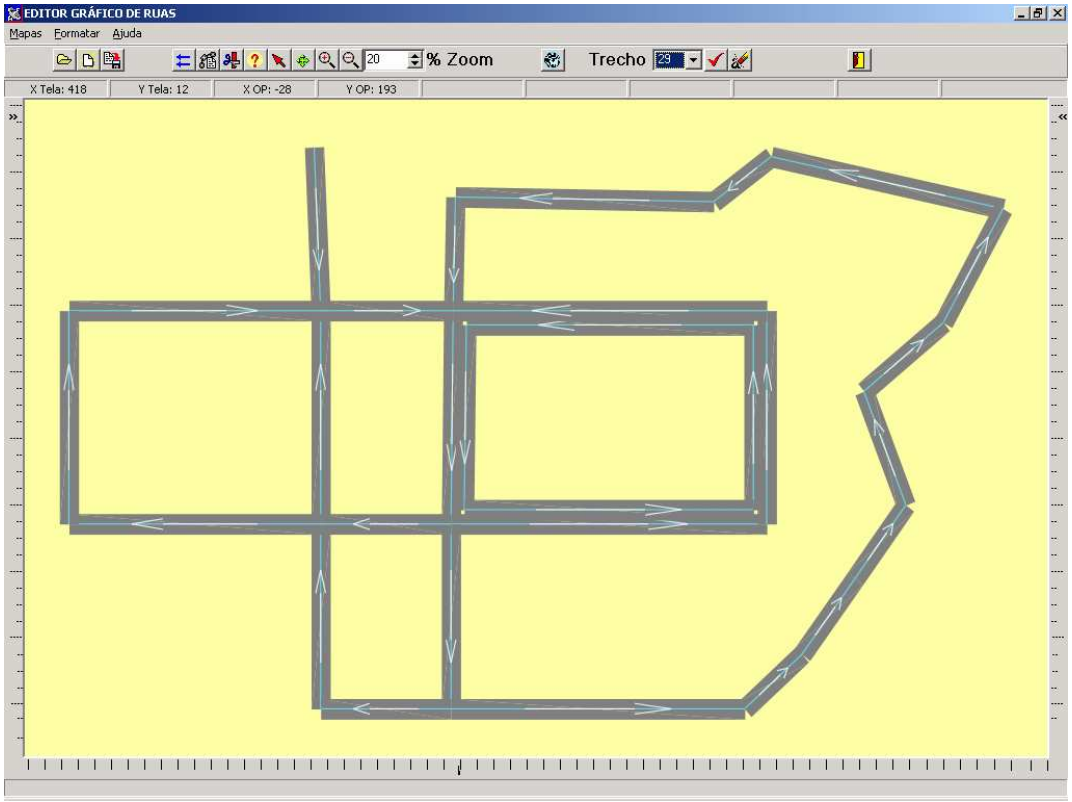
Rotina "DESENHAR TRECHO DO MAPA"

```
                              //  ROTINA DESENHA MAPA
Begin
  //verifica se malha não está vazia
  if L_Malha <> nil then
  begin
     //percorre a malha
     for z:= 0 to L_Malha.Count-1 do
     begin
        //aloca estrutura para manipulação
        Ponteiro:= L_Malha.Items[z];
        //chama a função desenhalinhalarg
        DesenhaLinhaLarg(Ponteiro.Inicio,Ponteiro.Fim,Ponteiro.F_Largura);
        //chama a função desenhalinhacentro
        DesenhaLinhaCentro(Ponteiro.Inicio,Ponteiro.Fim,Ponteiro.F_Largura);
        //chama a função desenhaseta
        DesenhaSeta(Ponteiro.Inicio, Ponteiro.Fim, Ponteiro.F_Largura);
     end;
  end;
End;                          //FIM DA ROTINA
```

Operacionalidade do Protótipo



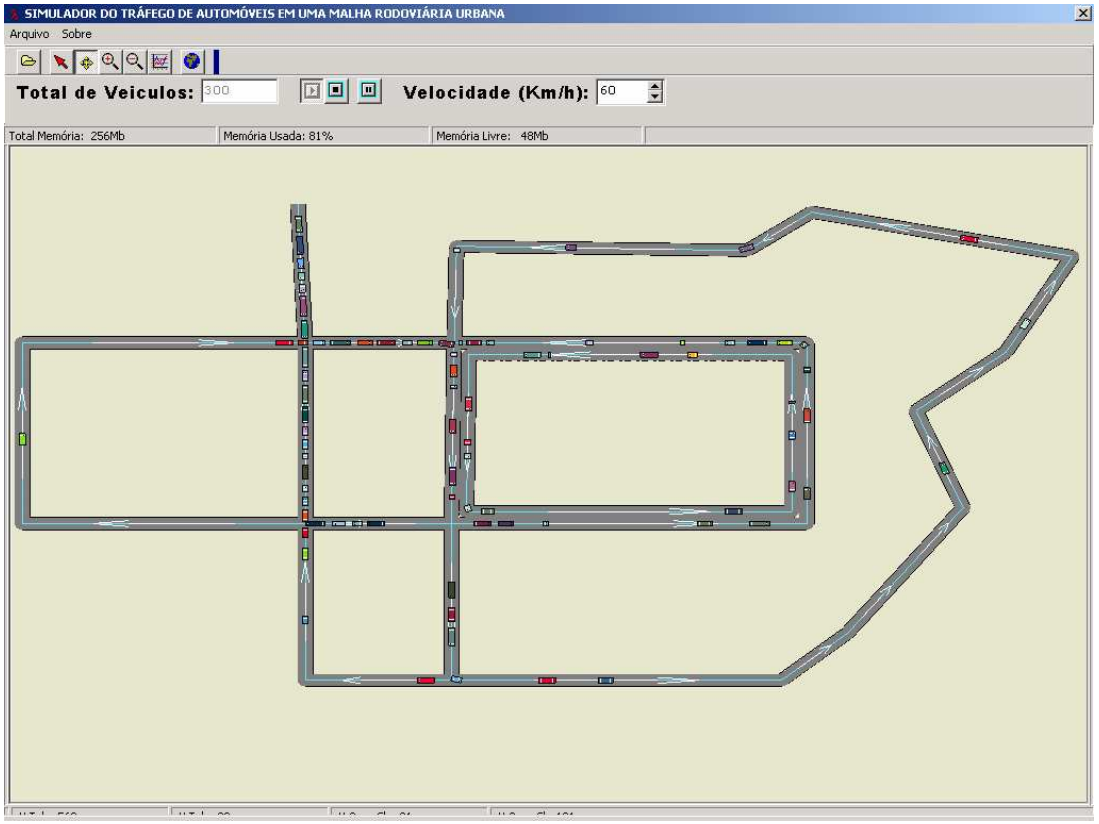
Resultados e Discussão



Resultados e Discussão

-94	179	-91	110	6	0	0	0	0	0	1
-30	158	-31	110	6	0	0	0	0	0	2
105	104	-25	104	6	18	0	0	0	0	3
-91	20	-91	110	6	0	0	0	0	0	4
-205	110	-91	110	6	0	0	0	0	0	5
-31	110	-32	20	6	19	0	0	0	0	6
-91	110	-31	110	6	0	0	0	0	0	7
-32	20	111	20	6	21	0	0	0	0	8
-91	-58	-91	20	6	0	0	0	0	0	9
-91	20	-205	20	6	0	0	0	0	0	10
-32	20	-32	-58	6	0	0	0	0	0	11
-32	20	-91	20	6	0	0	0	0	0	12
-205	20	-205	110	6	0	19	0	0	0	13
-32	-58	-91	-58	6	0	0	0	0	0	14
218	153	113	175	6	0	0	0	0	0	15
191	105	218	153	6	0	0	0	0	0	16
111	20	111	110	6	20	0	0	0	0	17
111	110	-31	110	6	0	3	0	0	0	18
-25	104	-26	26	6	0	6	0	0	0	19
105	26	105	104	6	0	17	0	0	0	20
-26	26	105	26	6	0	8	0	0	0	21
174	28	155	76	6	0	0	0	0	0	22
155	76	191	105	6	0	0	0	0	0	23
101	-58	127	-35	6	0	0	0	0	0	24
151	-3	174	28	6	0	0	0	0	0	25
127	-35	151	-3	6	0	0	0	0	0	26
87	156	-30	158	6	0	0	0	0	0	27
113	175	87	156	6	0	0	0	0	0	28
-32	-58	101	-58	6	0	0	0	0	0	29

Resultados e Discussão



Conclusão

Este trabalho apresentou o desenvolvimento de um editor gráfico de ruas para o sistema de controle de tráfego de automóveis. O ambiente de programação Delphi e a biblioteca gráfica OpenGl foram adequados para o desenvolvimento do protótipo, facilitando principalmente o desenho das primitivas geométricas.

As informações geradas pelo editor, quando salvas, são guardadas em arquivo texto compatível com o software desenvolvido por Jocemar J. Freire (FREIRE, 2004), visto que este trabalho trata-se de uma sugestão do trabalho desenvolvido pelo autor acima citado. A integração de ambos os softwares à partir do arquivo gerado funcionou.

Conclusão

O grande desafio do trabalho foi compreender inicialmente a biblioteca OpenGl para poder manipular as figuras matematicamente utilizando conceitos avançados de computação gráfica. A função ponto em polígono tornou-se extremamente vital para o protótipo, pela necessidade de localização do trecho no plano, pois a maioria das operações realizadas na aplicação em questão necessitam desta função.

Uma das principais dificuldades do uso da biblioteca OpenGl é a performance da aplicação à partir de um determinado número de trechos criados na malha, neste caso, é necessário alguns requisitos mínimos de *hardware*. Isto acontece porque as trechos são todos recalculados e redesenhados na tela à cada alteração feita pelo usuário.

Extensões

Para continuação deste trabalho são sugeridos as seguintes alterações:

- desenvolvimento de função para geração de trechos duplos ou triplos à partir de uma malha com trechos simples, facilitando ao usuário simular ampliações na malha viária;

- desenvolvimento de função que permita ao usuário o desenho da malha viária à partir de uma linha livre, ficando à cargo do sistema a transformação destas linhas em trechos retos (isto é possível através da utilização do algoritmo de Reumann-Witkam, segundo Queiroz (2004));

Extensões

- desenvolvimento de ferramenta para seleção de partes do desenho, objetivando a modificação das partes selecionadas ou a utilização da função copiar;
- desenvolvimento da função copiar/colar, possibitando ao usuário simplificar a criação de malha viária à partir de trechos semelhantes;
- desenvolvimento de função para criação de trechos à partir de informações como comprimento do trecho e angulação do trecho adjacente;