PROTÓTIPO DE SOFTWARE PARA GERAÇÃO DE ANIMAÇÕES POR QUADROS-CHAVES UTILIZANDO A TÉCNICA DE INTERPOLAÇÃO

ADRIANA DOS SANTOS

Dalton Solano dos Reis
Orientador

Roteiro

- Introdução
- Animação
- Tipos de animação
- Animação utilizando quadros-chaves
- Interpolação Quadros-Chaves
- Especificação / Implementação
- Conclusão / Extensões

Introdução

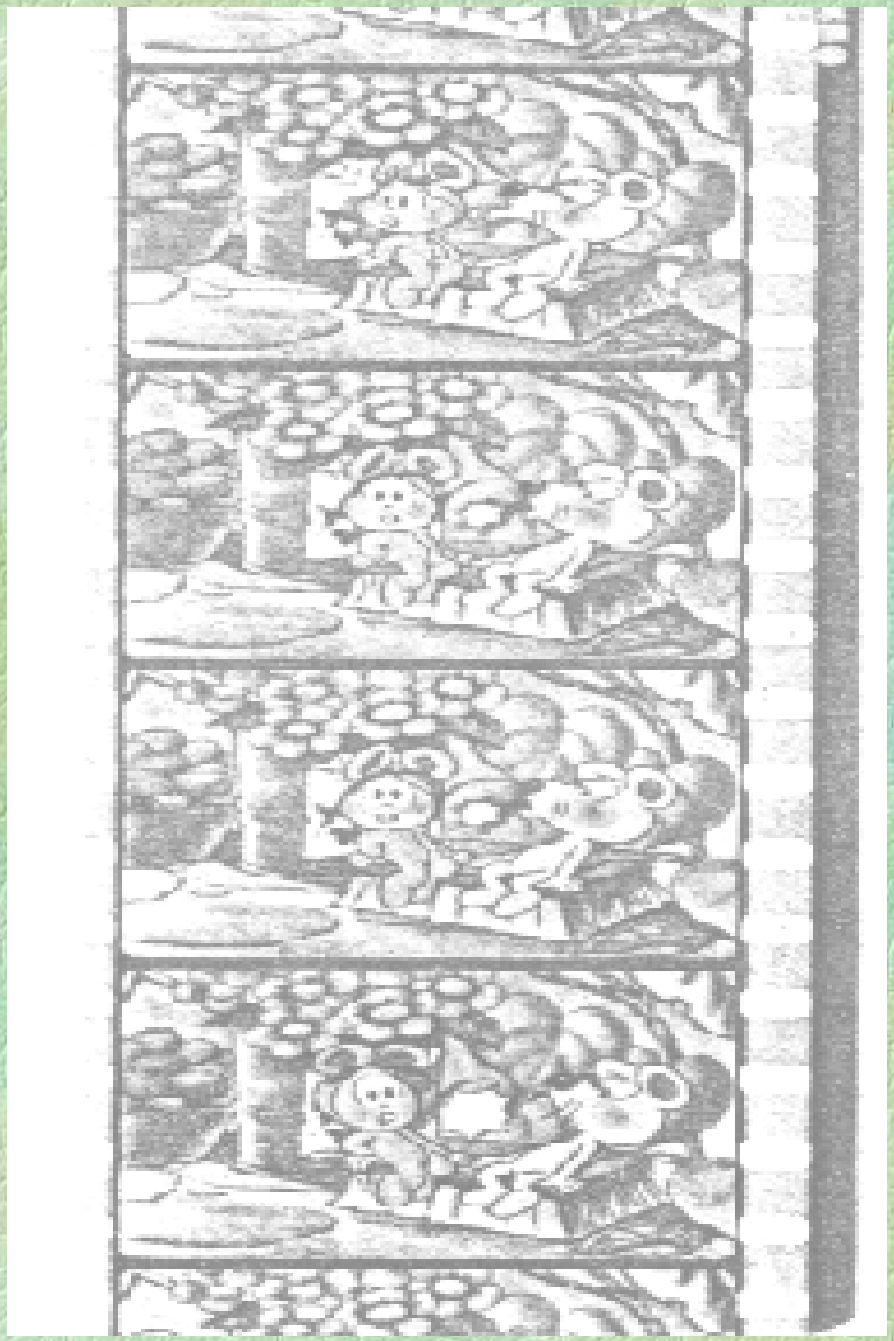
- Animação “precária”
- Benefícios do computador
- Técnicas de animação
- Animação por interpolação

Introdução - Objetivos

- Abordar evolução da animação
- Apresentar tipos de animação
- Técnica de animação - quadros-chaves
- Demonstrar Interpolação Matemática

Animação

- Movimento ilusório
- Olho humano: ilusão de movimento
- *Cel Animation*: planos de fundo
- Precursora: filmes, vinhetas, ...
- Walt Disney: Mickey Mouse (1923)



Animação
Projeção Imagens

Animação: uso do computador

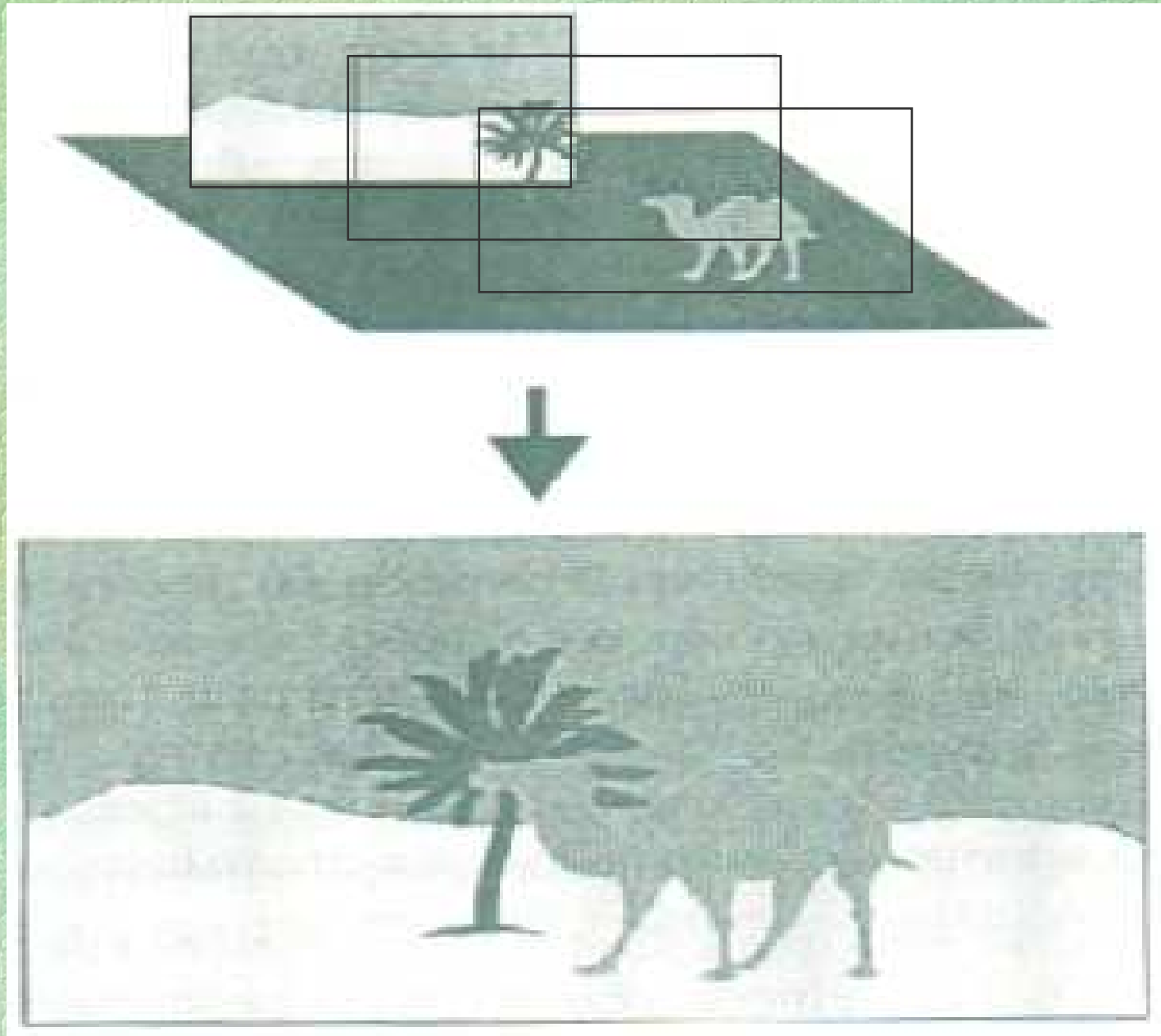
- Efeitos Especiais: ilusões fantásticas
- Talento Artístico: lápis, pincel e outros
- Artes Tridimensionais: esculturas, sombras
- Velocidade Projeção: qualidade imagem
- Excesso de utilização máquina
- Alterações: alterar, incluir, modificar

Tipos de Animação

- **Tradicional**
- BITBL
- por *Scripts*
- por Deslocamento
- *Scan*
- Bidimensional (2D)
- Tridimensional (3D)
- Tempo Real e Simulado
- **por Quadros-Chaves**

Tipos de Animação: Tradicional

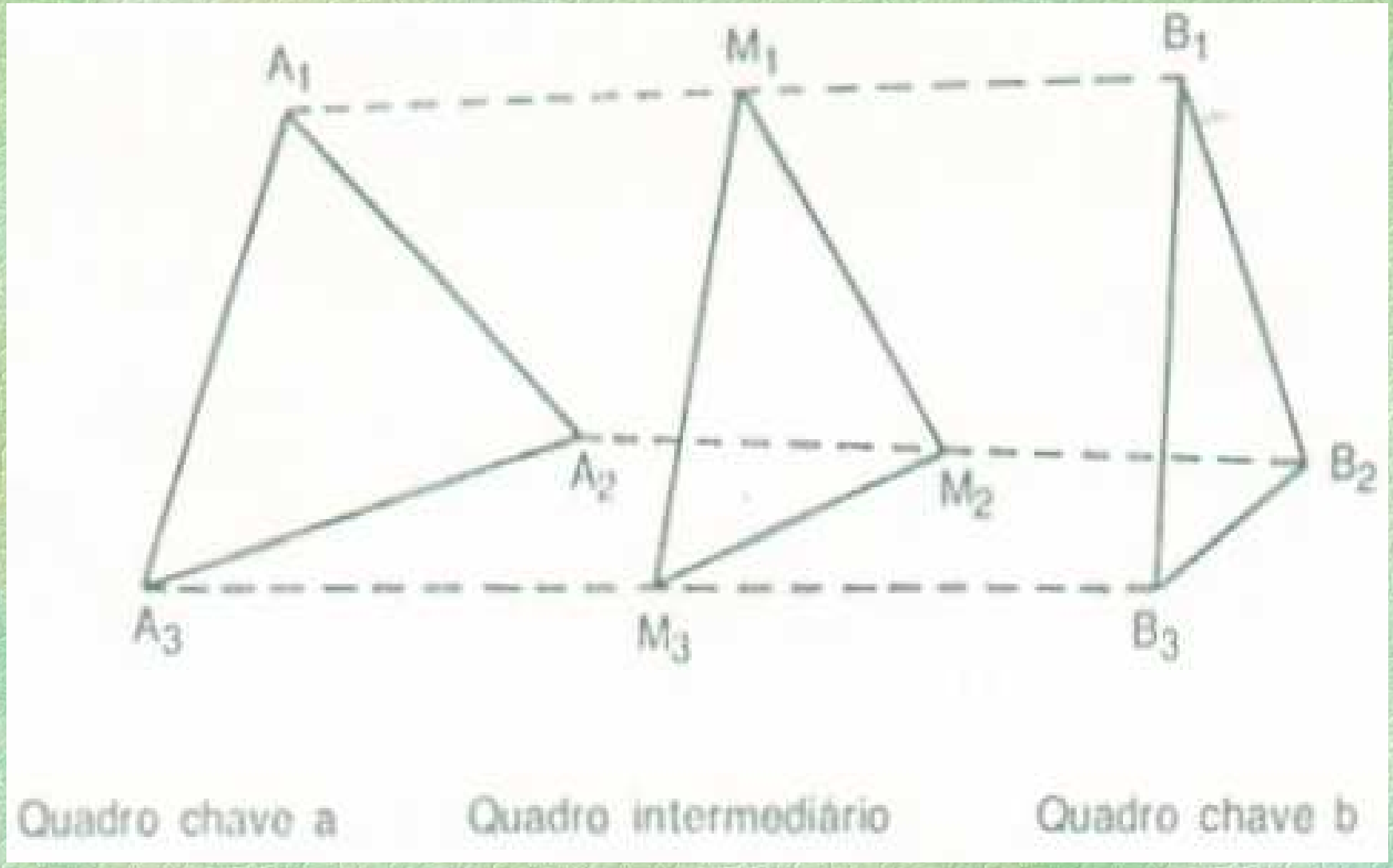
- Equipe Artistas: esboços, cultura, idioma
- *Storyboard*: definição história, detalhes
- Cenários: *layout*, planos de fundo
- Câmera: registra com liberdade
- Camadas: deslocamentos, movimentos



Planos de Fundo

Tipos de Animação: Quadros-Chaves

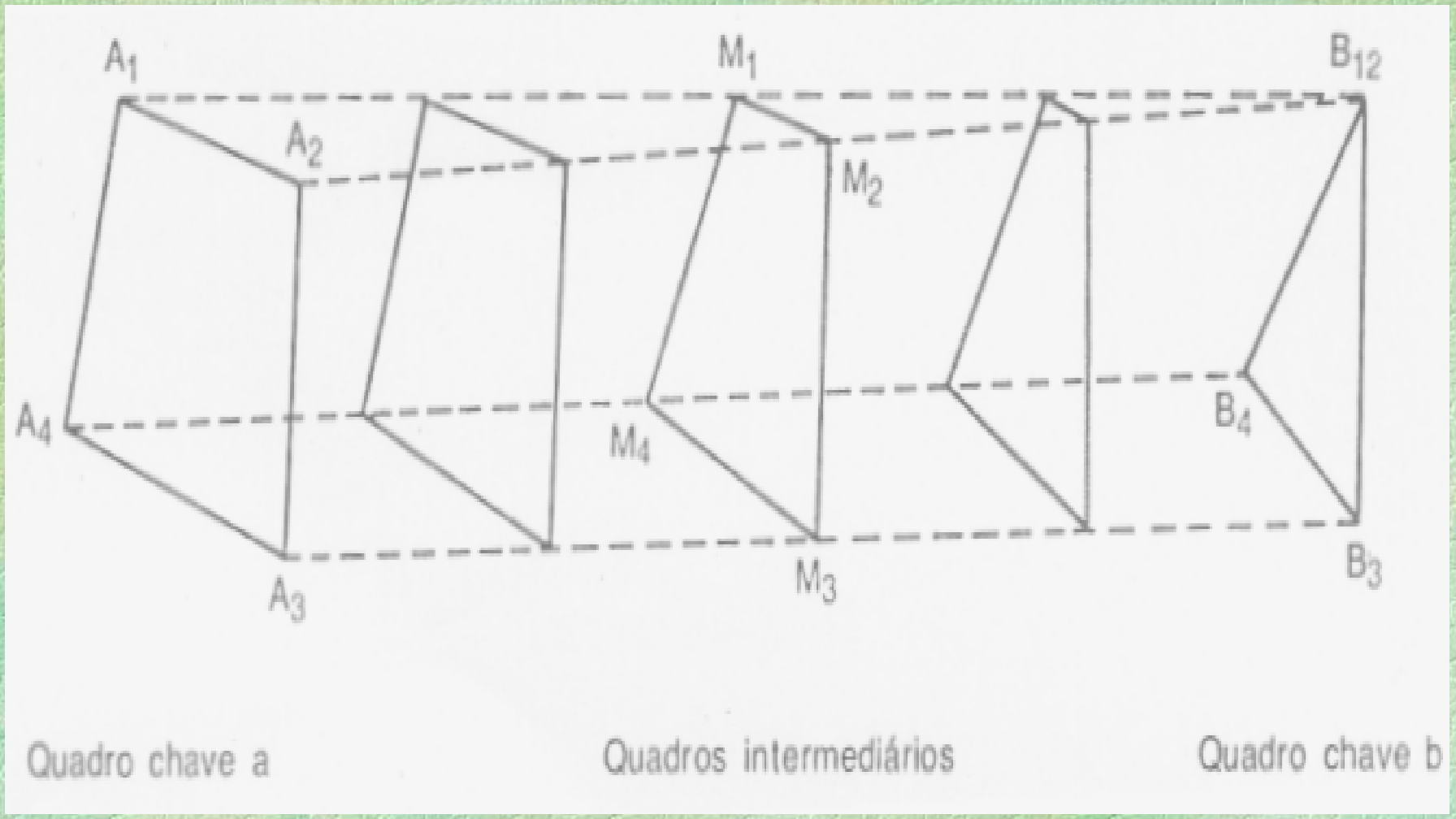
- Processo: desenhos estáticos, trabalhoso
- Criação: pontos “Chaves” Início / Fim
- Continuidade: quadros intermediários
- Forma: interpolação



Quadros Chaves/Intermediários

Interpolação Quadros-Chaves

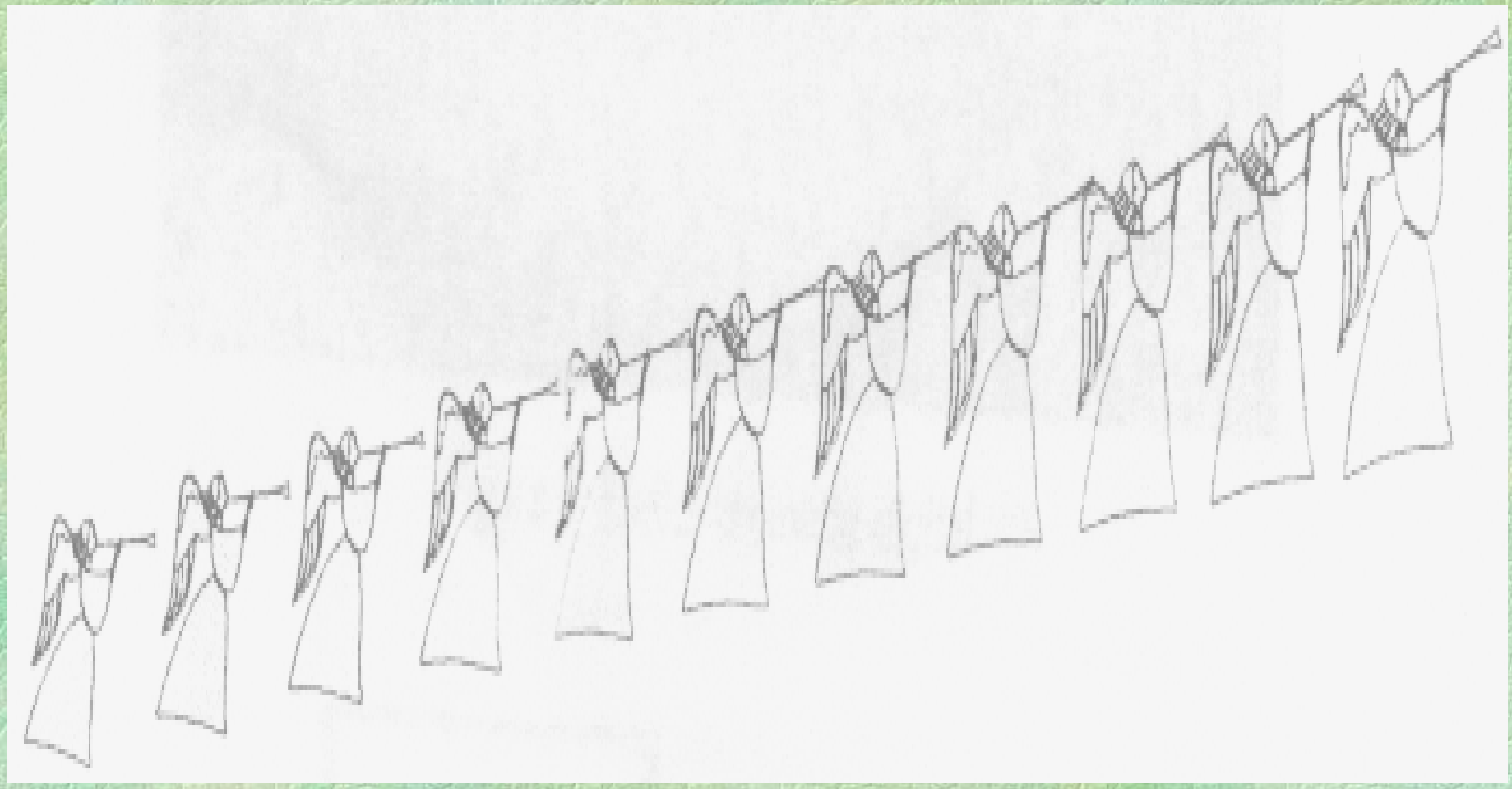
- Método: técnica continuidade movimento
- Interpolação: pontos correspondentes
- Quadros: Origem → Destino → Origem
- Fórmulas:
  - transformações geométricas,
  - Matrizes,
  - entre outros



Quadros origem / destino

Interpolação - Dificuldades

- Criação novos pontos
- Difusão de pontos em um ponto
- Interface Usuário: visualização
- Animação Natural: saltos, tempo



Animação Contínua

Especificação Protótipo

- Prototipação Fundamental: Evolutivo
  Produto Final → Software

- Ambiente Delphi 4.0 - componentes
- Diagrama de Contexto
- Diagrama de Fluxo de Dados
- Fluxograma

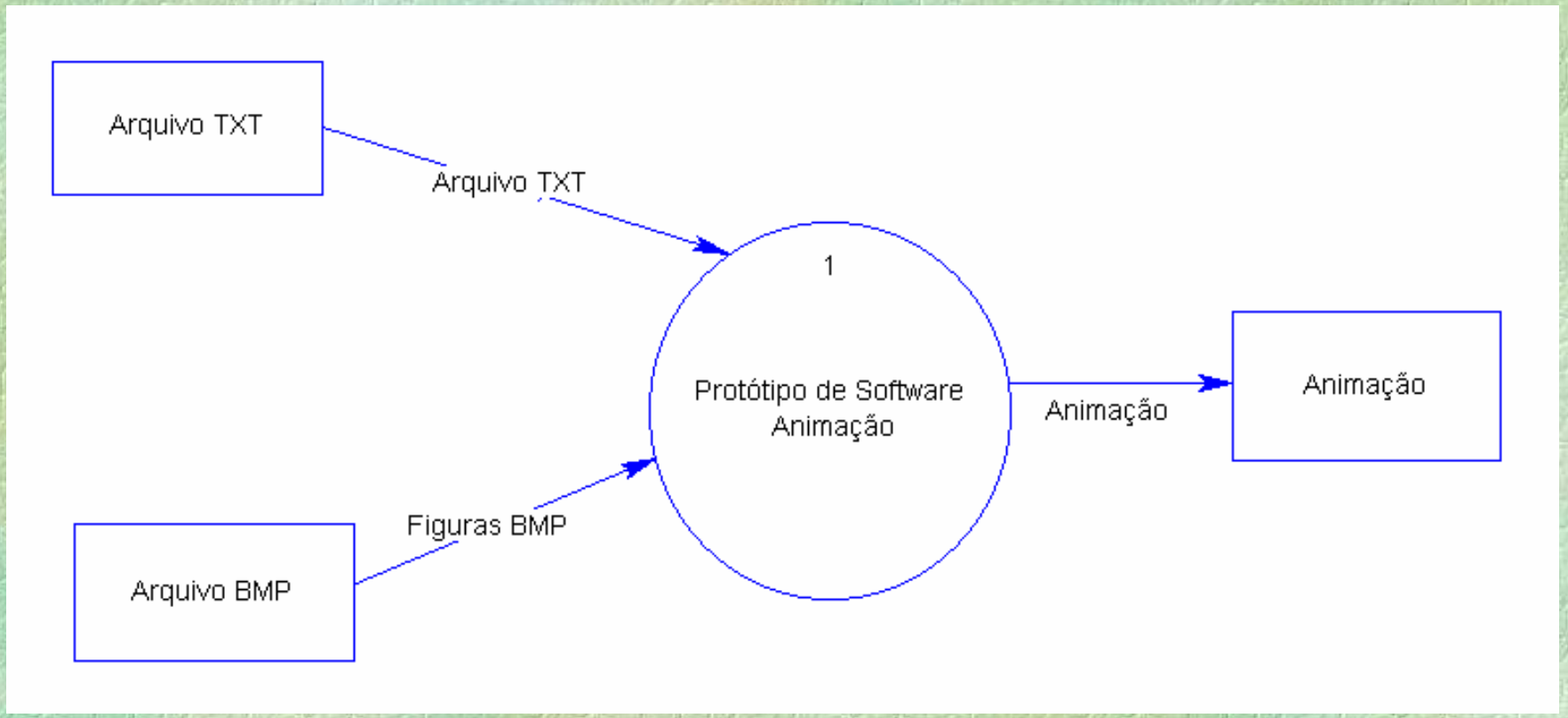


Diagrama de Contexto

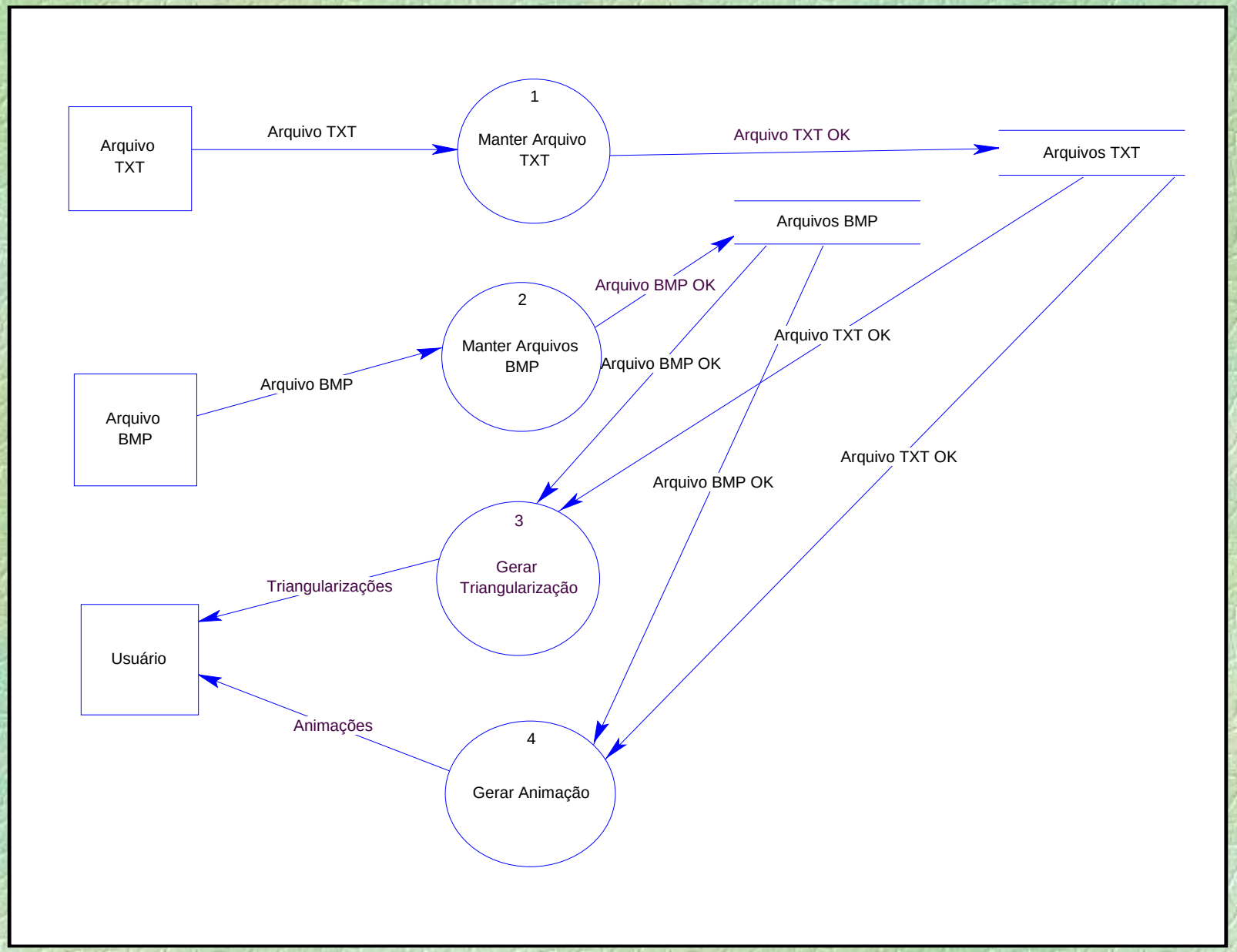
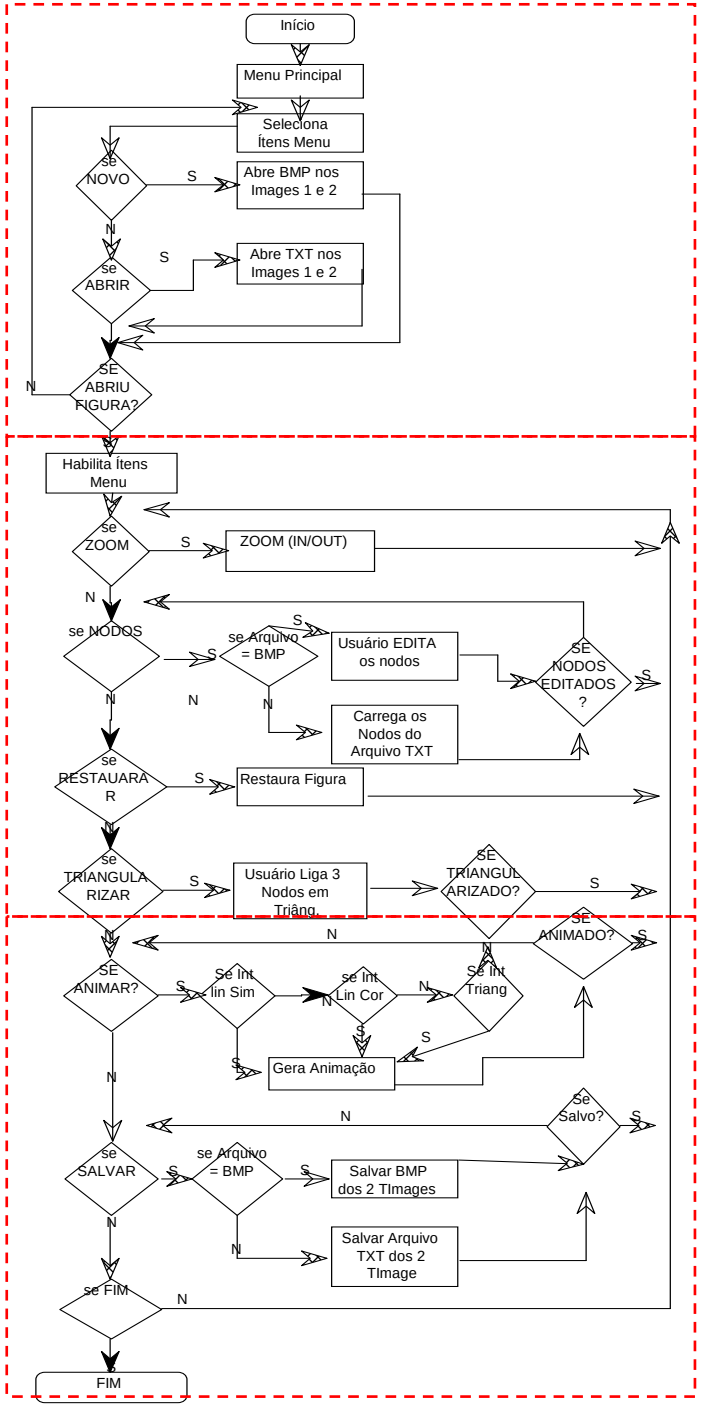
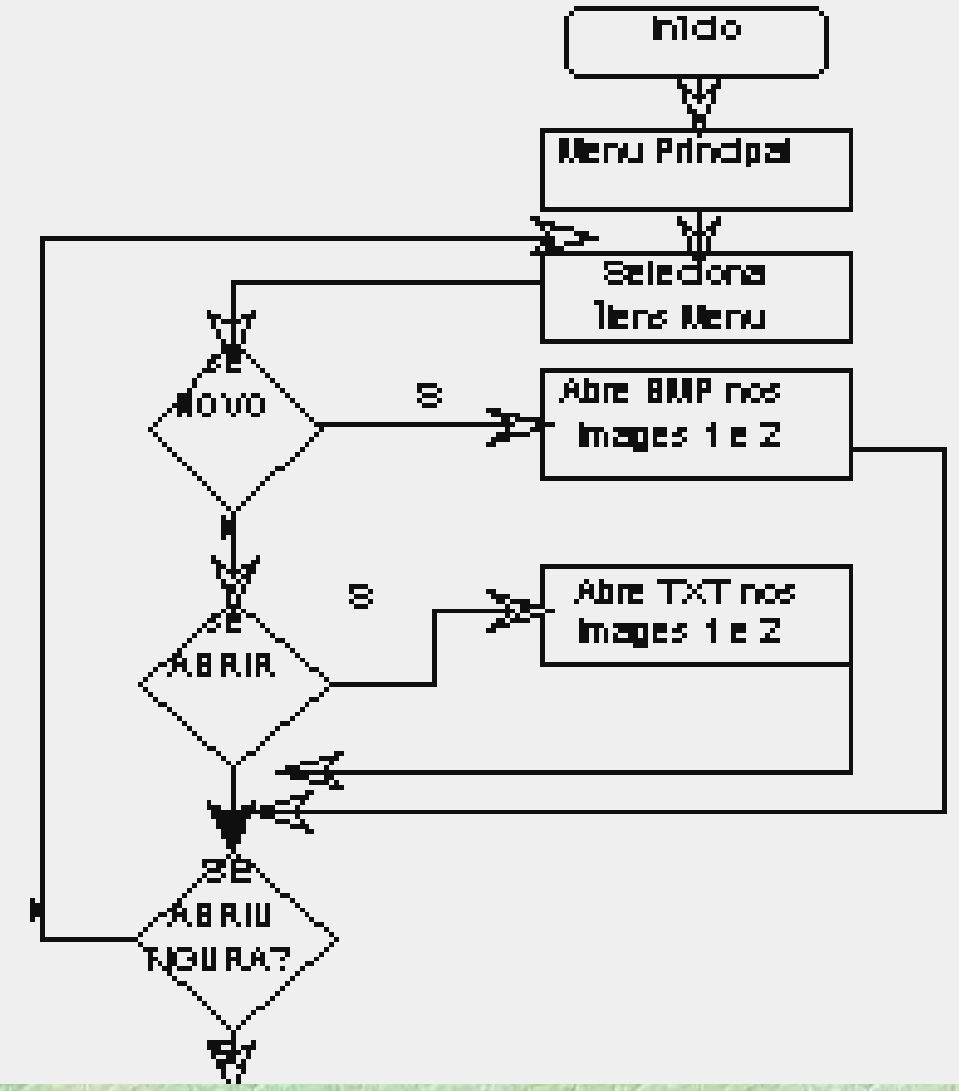


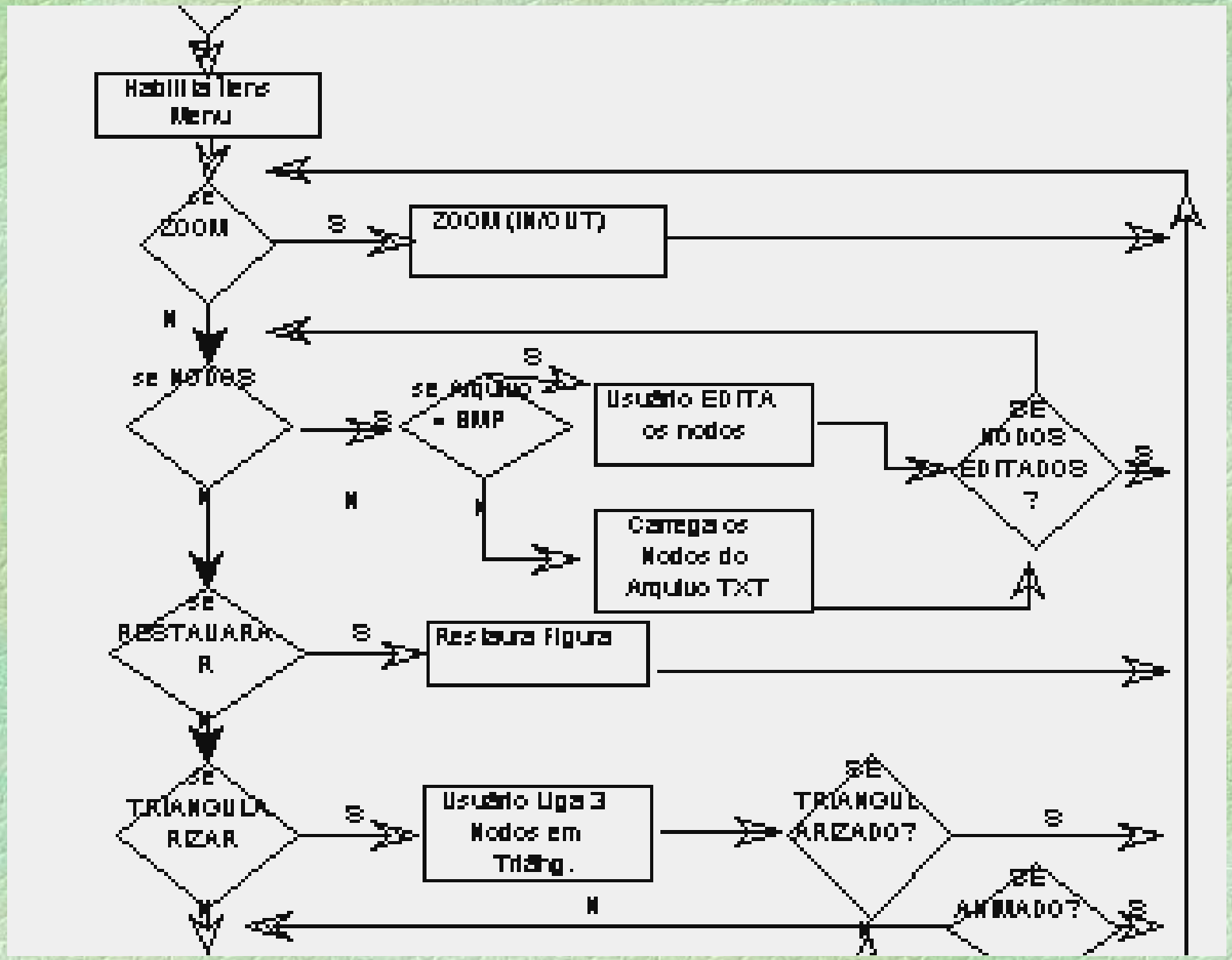
Diagrama de Fluxo de Dados - DFD


Início
Menu Principal
Seleciona Ítens Menu
se NOVO
S
Abre BMP nos Images 1 e 2
N
se ABRIR
S
Abre TXT nos Images 1 e 2
SE ABRIU FIGURA?
N
Habilita Ítens Menu
se ZOOM
S
ZOOM (IN/OUT)
N
se NODOS
S
se Arquivo = BMP
S
Usuário EDITA os nodos
SE NODOS EDITADOS ?
S
N
N
Carrega os Nodos do Arquivo TXT
se RESTAUARAR
S
Restaura Figura
se TRIANGULARIZAR
S
Usuário Liga 3 Nodos em Triâng.
SE TRIANGULARIZADO?
S
N
SE ANIMADO?
S
N
N
SE ANIMAR?
S
Se Int lin Sim
se Int Lin Cor
N
Se Int Triang
N
S
S
S
Gera Animação
N
Se Salvo?
S
N
se SALVAR
S
se Arquivo = BMP
S
Salvar BMP dos 2 TImages
N
N
Salvar Arquivo TXT dos 2 TImage
se FIM
N
FIM

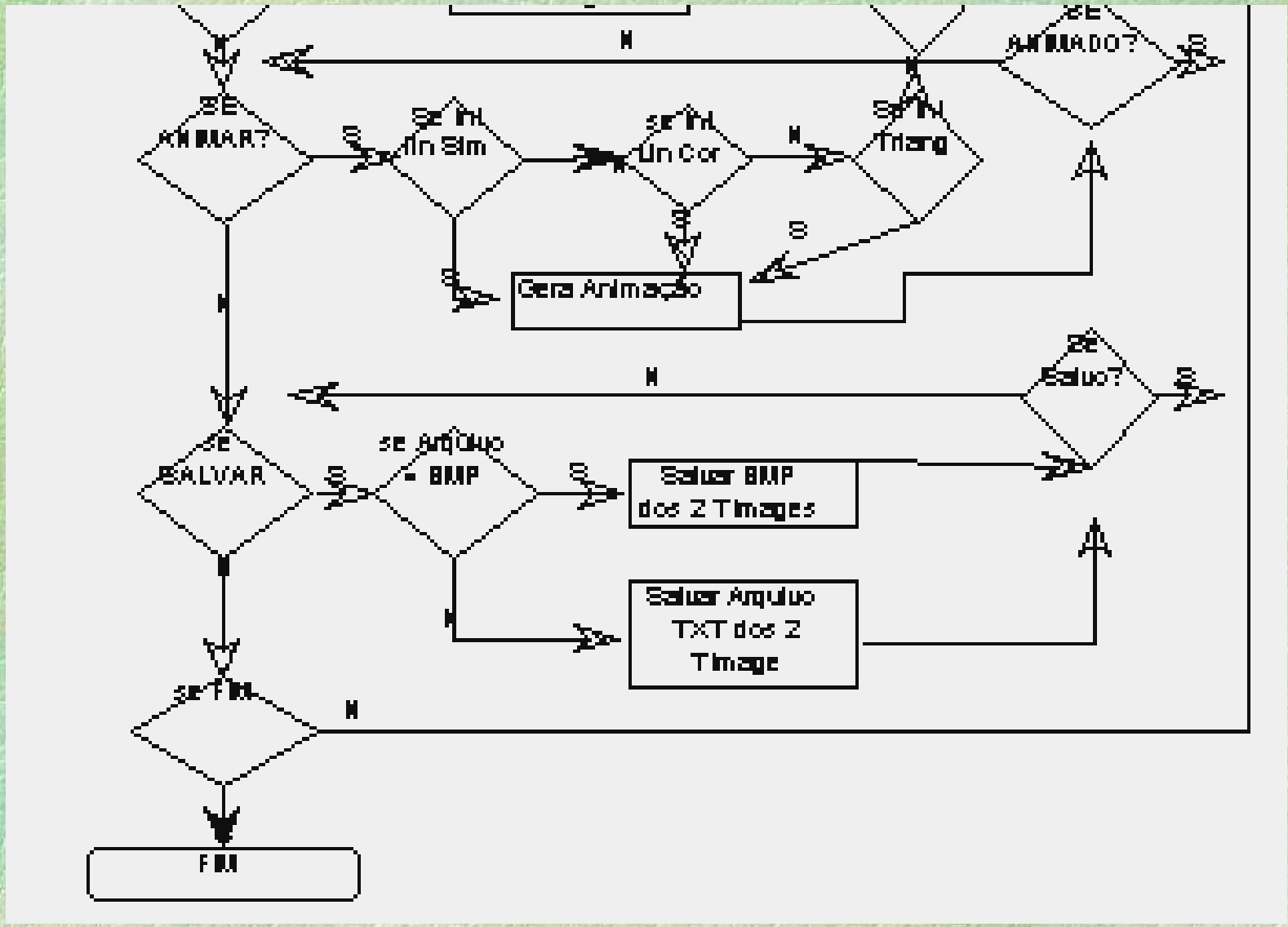
Fluxograma


Início
Menu Principal
Selecione
Item Menu
É
NOVO
S
Abre BMP nos
Images 1 e 2
É
ABRIR
S
Abre TXT nos
Images 1 e 2
SE
ABRIU
FIGURA?

Fluxograma


Habilita Itens Menu
SE ZOOM
S
ZOOM (IN/OUT)
N
se MODOS
S
se Arquivo = BMP
S
Usuário EDITA os nodos
SE NODOS EDITADOS ?
S
N
N
Carrega os Nodos do Arquivo TXT
SE RESTAURAR
S
Restaura Figura
SE TRIANGULAR
S
Usuário Liga 3 Nodos em Triâng.
SE TRIANGULARIZADO?
S
N
SE ANULADO?
S

Fluxograma


N
S
Gera Animação
Salvar BMP
dos 2 Timages
Salvar Arquivo
TXT dos 2
Timage
FIM

Fluxograma

Implementação Protótipo

- Arquivos *Script*
- Interpretação comandos: *Tmemo*
- Desenho Figuras: *zoom in / out*
- Redimensionamento:
  - Multiplicação / Divisão (constante 2)

Gerando Animação

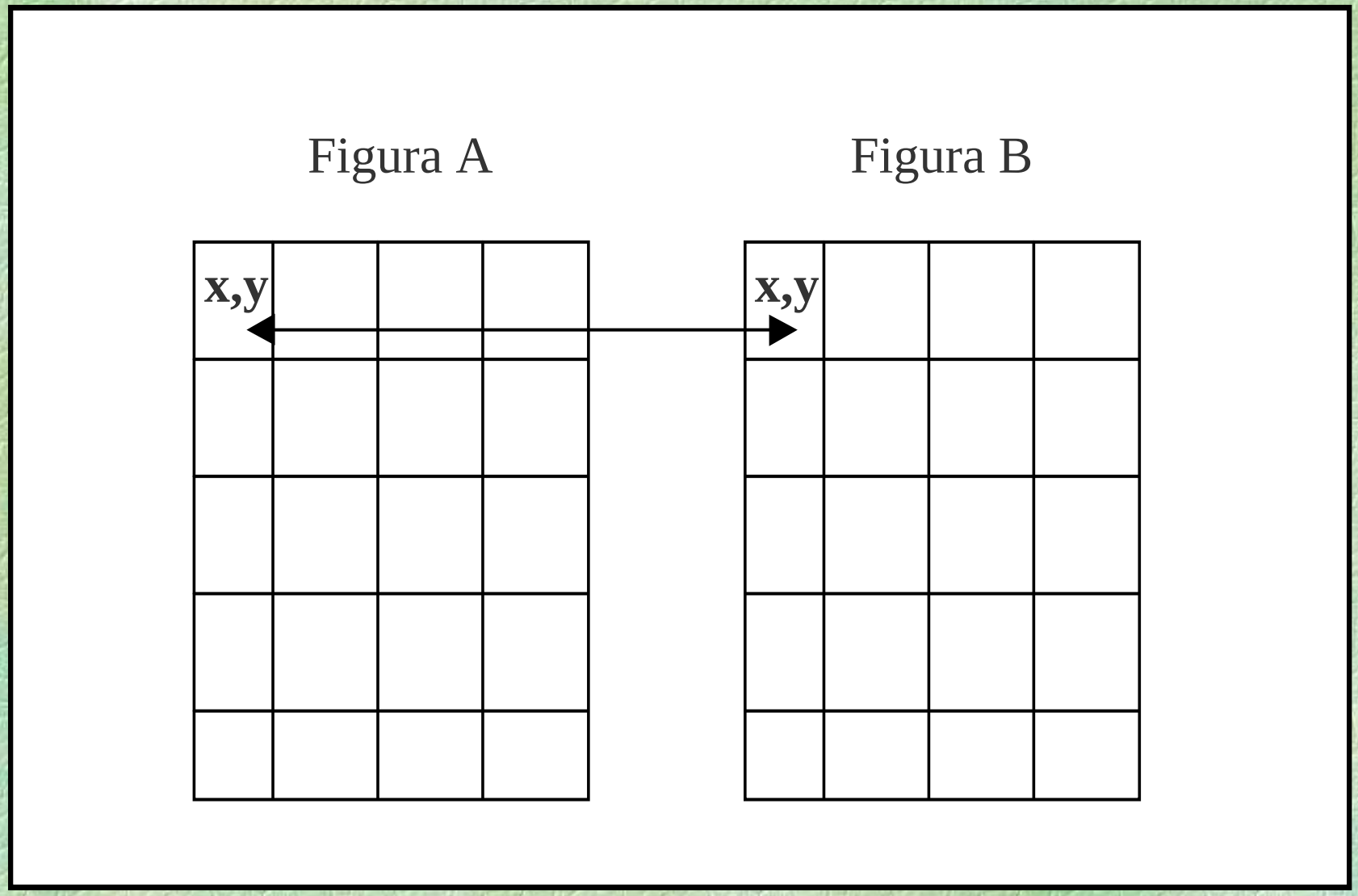
- Interpolação:
  - Linear Ponto-a-Ponto
  - Linear Ponto-a-Ponto RGB
  - Linear pontos Médios
- Equação Paramétrica
- Equação Pontos Médios

Linear Ponto-a-Ponto

- Passo 1: varre-se figura A
- Passo 2: interpola-se *Pixeis* entre A e B
- Passo 3: Repetição: μ vezes [0..1]

Interpolação: Equação Paramétrica

$$Qi = A + (B - A) * \mu$$



Coordenadas - entre A e B

Linear Ponto-a-Ponto RGB

- “Mesma” definição do Linear Ponto-a-Ponto
- Interpolação:
  - Passo 1: captura a cor do *pixel*
  - Passo 2: “particiona” *pixel* (RGB)
  - Passo 3: equação paramétrica
    - interpolar R
    - interpolar G
    - Interpolar B
  - Passo 4: “juntar” *pixel*

Linear Pontos Médios

- Edição: pontos – triângulos
- Gerar Quadros: μ [0.. 1]
  - Para cada triângulo:
    - Calcula valores Intermediários (pontos / cor)
    - Calcula valores Médios (pontos / cor)
    - Processo Recursivo - Calcula valores Médios
  - Próximo Triângulo
- Próximo Quadro

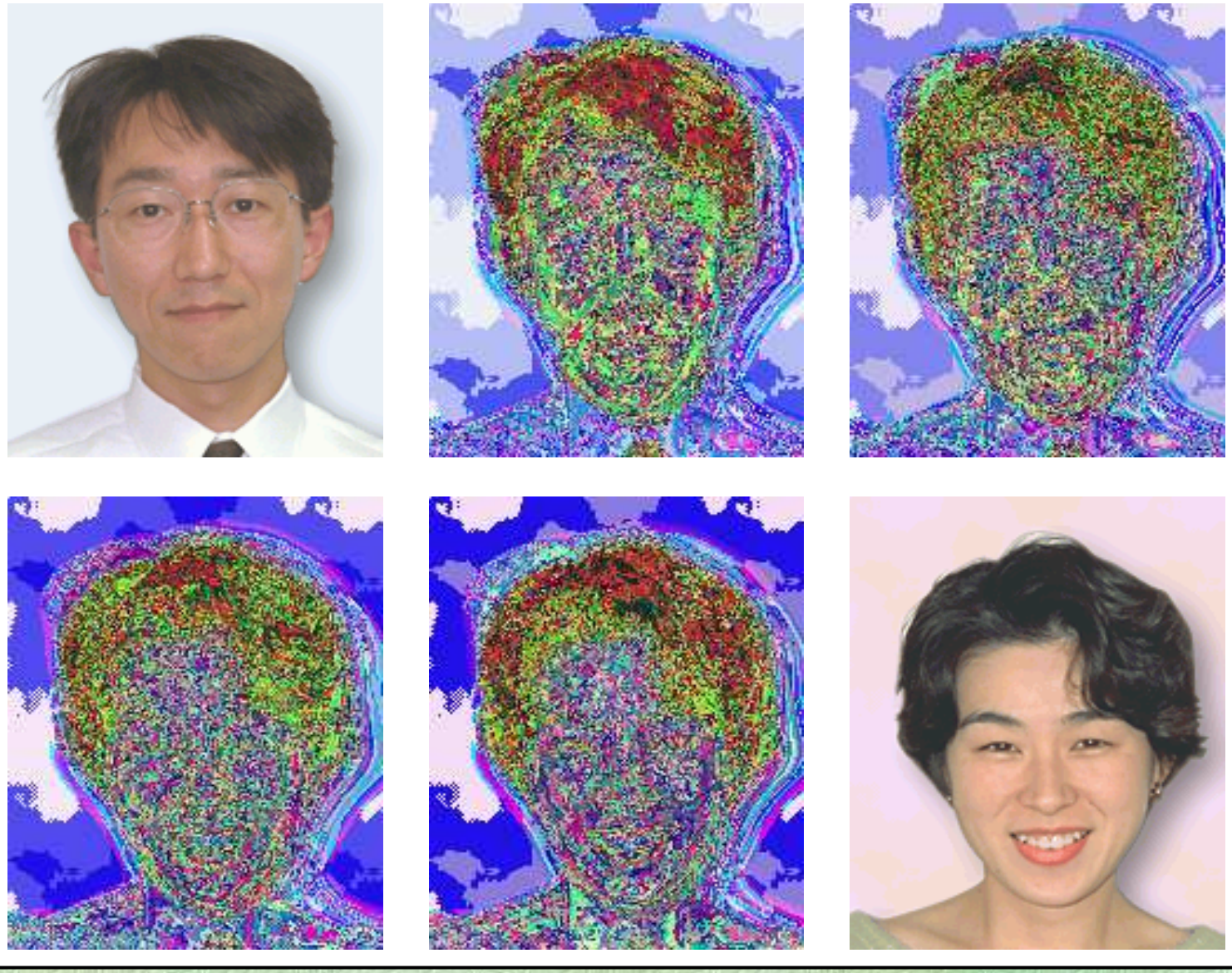
Linear Pontos Médios

- Equação Paramétrica - valores Intermediários

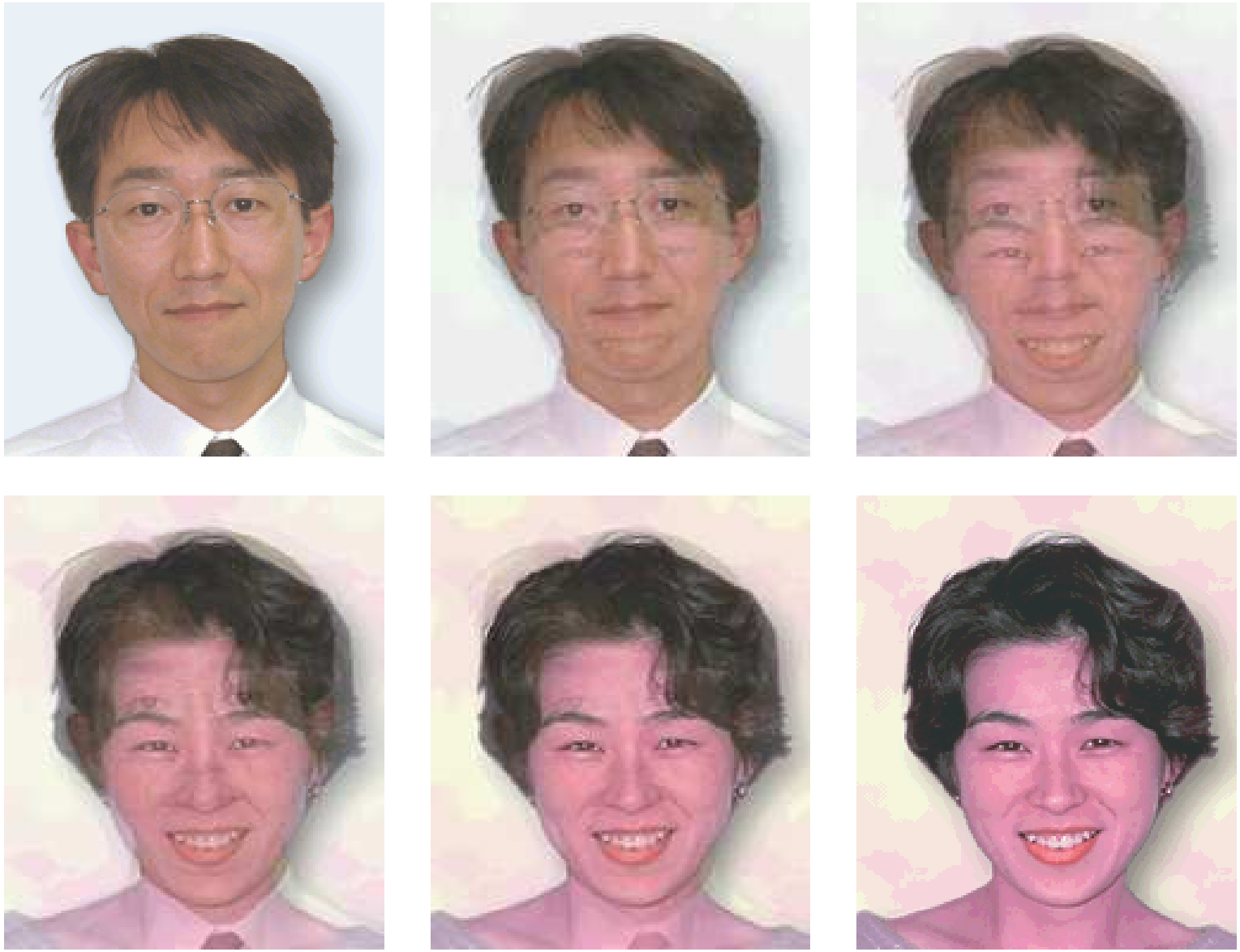
$$Qi = A + (B - A) * \mu$$

- Equação Ponto Médio - valores Médios

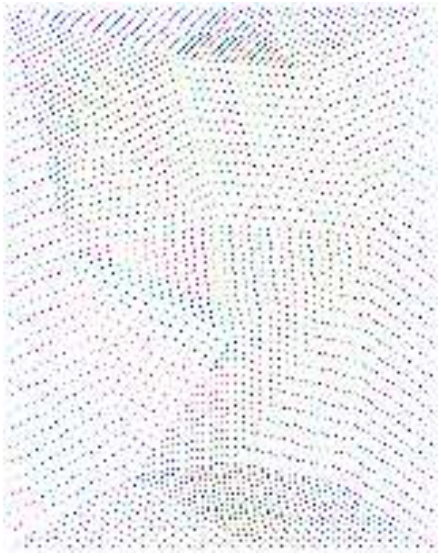
$$Pm = (A + B) / 2$$

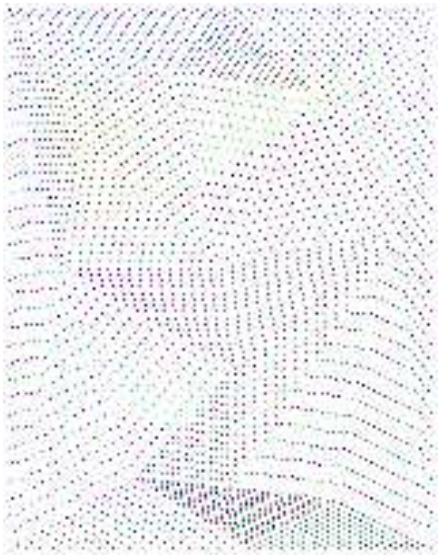


Linear Ponto-a-Ponto



Linear Ponto-a-Ponto RGB


Linear Pontos Médios

Conclusão

- Primeiro Método:
  - “mais” intuitivo,
  - menor proximidade animação
- Segundo Método:
  - extensão primeiro,
  - maior proximidade,
  - relevância da Cor (*pixeis*)
- Terceiro Método:
  - sobrecarga devido recursividade
  - baixo aspecto visual

várias extensões

Extensões

- Demais métodos mencionados
- Tratamento de arquivos vetoriais
- Tratamento de *pallet's* das imagens
- Terceiro Método: implementar com RGB
- Aumentar o grau de recursividade
- Implementar algoritmos triangularização
- Janela para valores gerados (depuração)

Apresentação do Protótipo

Zoom In

```
if ( FmNovo.LarguraFig1 <> FmNovo.LarguraFig2) or
     ( FmNovo.AlturaFig1  <> FmNovo.AlturaFig2 ) then
  begin
      ShowMessage ( ' Tamanho das Figuras Diferentes' + #13#10 +
                    ' A Primeira Figura Tem de Largura: '     +
                      FloatTostr(FmNovo.LarguraFig1) + #13#10 +
                    ' A Primeira Figura Tem de Altura:  '     +
                      FloatTostr(FmNovo.AlturaFig1)  + #13#10 +
                    ' A Segunda  Figura Tem de Largura: '     +
                      FloatTostr(FmNovo.LarguraFig2) + #13#10 +
                    ' A Segunda Figura Tem de Altura :  '     +
                      FloatTostr(FmNovo.AlturaFig2)  );
      FmNovo.Close;
  end
  else
  begin
     FmNovo.Width := trunc(FmNovo.LarguraFig1) +
                     trunc(FmNovo.LarguraFig2) + 45;
     FmNovo.Height:= trunc(FmNovo.AlturaFig1)  + 45;
     FmNovo.Image1.Picture.LoadFromFile(FmNovo.NomeFigura1);
     FmNovo.Image2.Picture.LoadFromFile(FmNovo.NomeFigura2);
     Fmnovo.Image1.Left:=0;
     FmNovo.Image1.top:=0;
     Fmnovo.Image2.top:=0;
     Fmnovo.Image2.left:= trunc(FmNovo.LarguraFig1) + 10;
     FmNovo.Show;
  end;   {  Para as Figuras Iguais }
```