Departamento de Sistemas e Computação – FURB
Curso de Ciência da Computação
Trabalho de Conclusão de Curso – 2018/1

CORPUSVR: Um protótipo para visualização da anatomia do corpo humano

Acadêmico: Danilo Conrado Eskelsen Junior
danilo_cej@hotmail.com

Orientador: Prof. Aurélio Faustino Hoppe
aurelio.hoppe@gmail.com

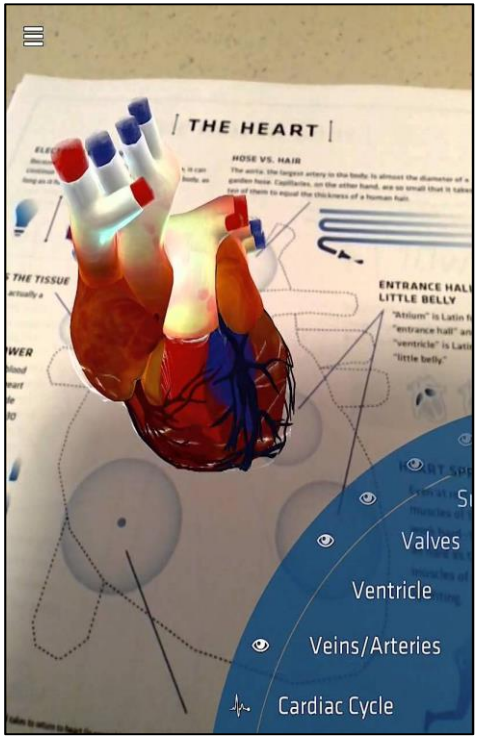
Grupo de Processamento de Imagens, Robótica e Simulação computacional

Roteiro

- Introdução
- Objetivos
- Fundamentação teórica
- Trabalhos correlatos
- Requisitos
- Ferramentas
- Implementação
- Resultados
- Conclusões
- Extensões

Introdução

- Para estudar as estruturas anatômicas ainda é muito comum o uso de aulas expositivas, atlas e a dissecação
- O uso da RA com um smartphone pode ajudar na compreensão e visualização de partes do corpo humano



Objetivos

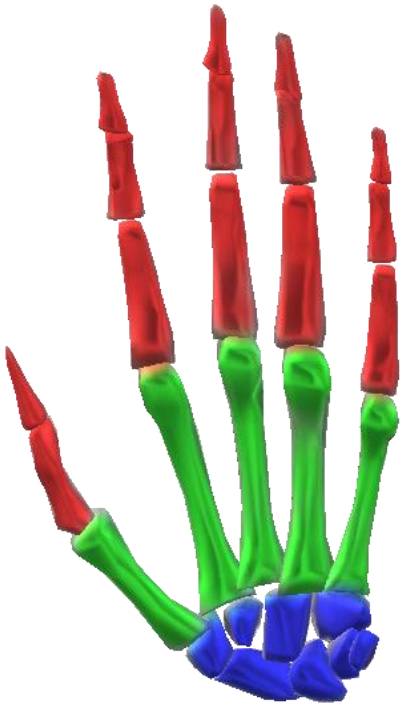
Desenvolver uma aplicação fazendo uso da tecnologia de RA para visualização de parte da anatomia do corpo humano em tempo real

Objetivos específicos:

I. identificar o contorno da mão a partir da segmentação da imagem com base na cor da pele
II. estimar a pose da mão a partir dos principais pontos convexos do contorno
III. renderizar o modelo esquelético 3D utilizando a pose estimada da mão
IV. otimizar o desempenho do módulo da câmera da biblioteca OpenCV

Fundamentação teórica (1/3)

- O sistema esquelético da mão é subdivido em 3 regiões
- falange: 14 ossos que compreendem os dedos da mão e se articulam com os ossos do metacarpo
- metacarpo: 5 ossos que se articulam com os ossos do carpo e com as falanges
- carpo: 8 ossos que compreendem a região do pulso



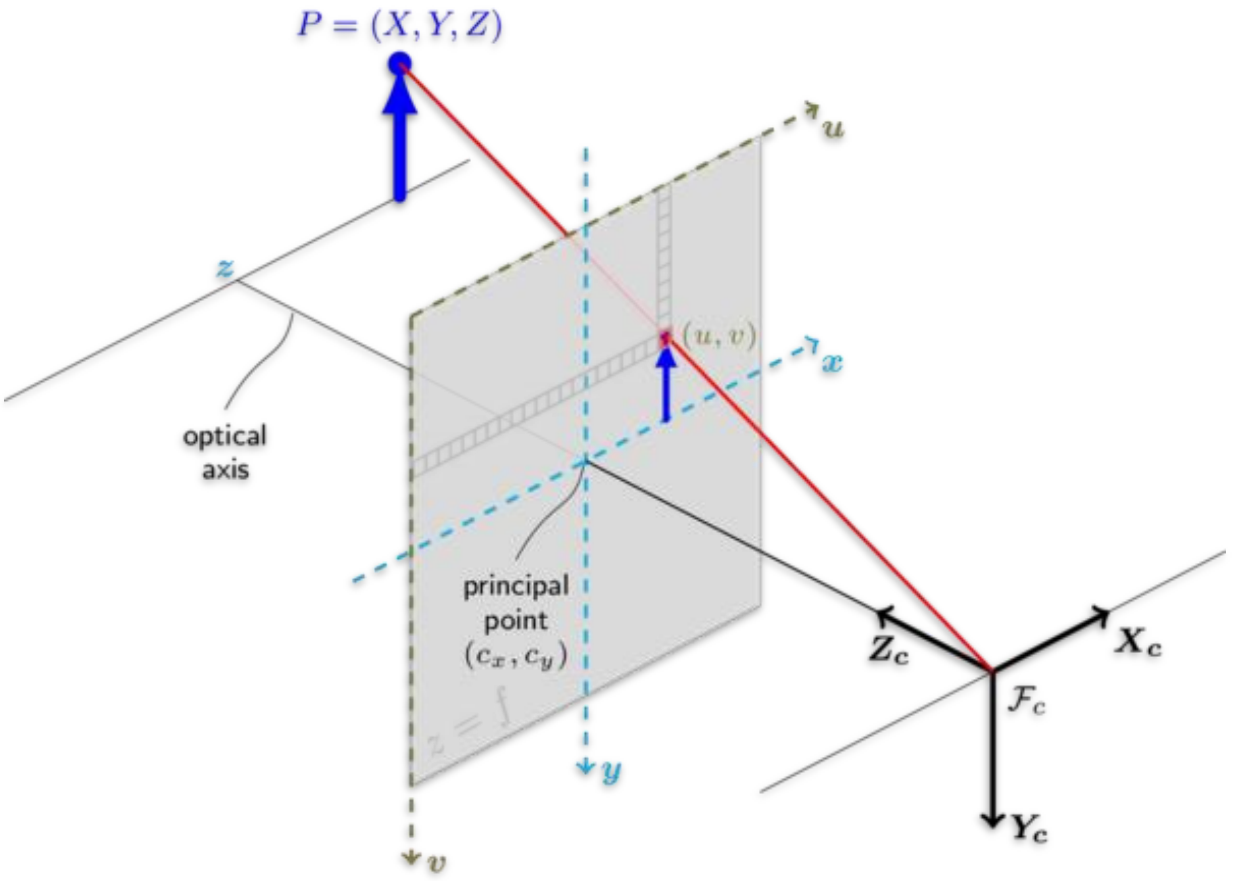
Fundamentação teórica (2/3)

- Matrizes de transformações do OpenGL: translação, escala, rotação e identidade

$$\begin{bmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \\ m & n & o & p \end{bmatrix} \cdot \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{pmatrix} a \cdot x + b \cdot y + c \cdot z + d \cdot 1 \\ e \cdot x + f \cdot y + g \cdot z + h \cdot 1 \\ i \cdot x + j \cdot y + k \cdot z + l \cdot 1 \\ m \cdot x + n \cdot y + o \cdot z + p \cdot 1 \end{pmatrix}$$

Fundamentação teórica (3/3)

- Modelo de câmera *pinhole*

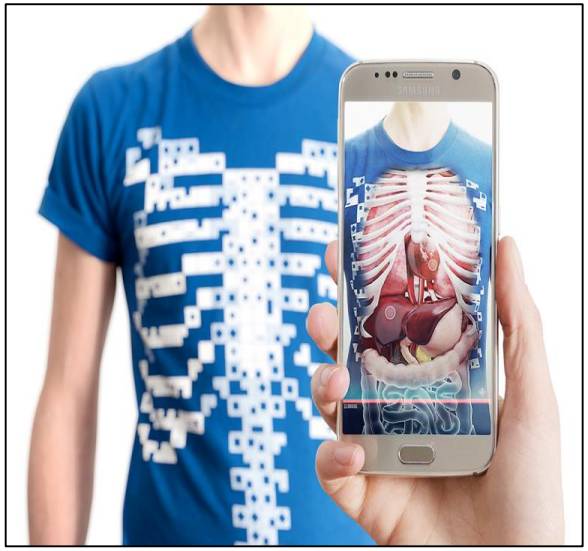


Trabalhos correlatos (1/3)

Título: Virtuali-Tee

Curiscope (2016)

características / trabalhos relacionados	Curiscope (2016)
Plataforma	Android / iOS
Faz uso de cartão de RA	✗
Permite interação com a animação	✓
Anatomia completa do corpo humano	✗
Possui modo de visualização em RV (cardboard)	✓

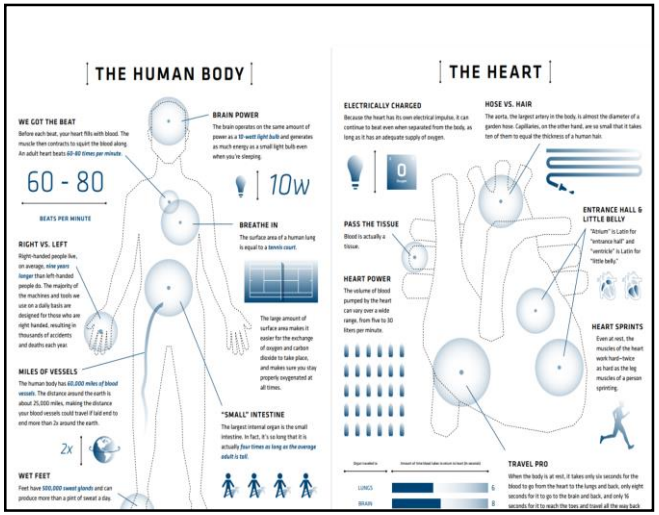


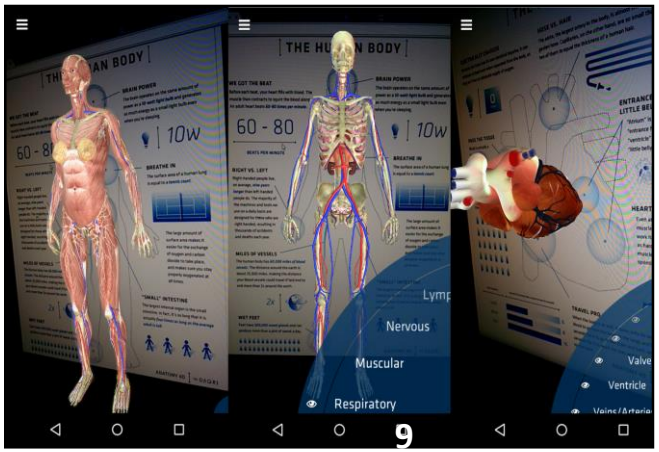
Trabalhos correlatos (2/3)

Título: Anatomy 4D

Daqri (2014)

características / trabalhos relacionados	Daqri (2014)
Plataforma	Android / iOS
Faz uso de cartão de RA	✓
Permite interação com a animação	✓
Anatomia completa do corpo humano	✓
Possui modo de visualização em RV (cardboard)	✗





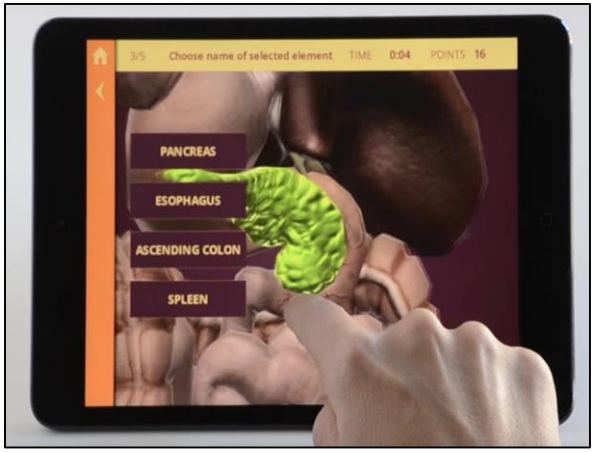
9

Trabalhos correlatos (3/3)

Título: Arloon Anatomy

Arloon (2015)

características / trabalhos relacionados	Arloon (2015)
Plataforma	Android / iOS
Faz uso de cartão de RA	✓ ✗
Permite interação com a animação	✓
Anatomia completa do corpo humano	✓
Possui modo de visualização em RV (cardboard)	✗



Requisitos

requisito	descrição
RF01	permitir que o usuário utilize a câmera do dispositivo para obter as imagens
RF02	segmentar uma imagem com base na cor da pele
RF03	identificar o maior contorno de uma imagem segmentada
RF04	identificar os pontos convexos de um contorno
RF05	identificar os principais pontos convexos
RF06	estimar a pose da mão a partir de uma lista de pontos convexos
RF07	permitir ao usuário de visualizar o modelo 3D do sistema esquelético da mão
RF08	imitar a pose da mão com o modelo 3D do sistema esquelético
RNF01	ser desenvolvida para a plataforma Android
RNF02	utilizar o ambiente de desenvolvimento Android Studio
RNF03	utilizar a biblioteca de visão computacional OpenCV para fazer o processamento de imagem
RNF04	utilizar o motor gráfico Rajawali para renderizar o sistema esquelético da mão

Ferramentas utilizadas

Implementação

- **Thread 1**: identificação da mão
- **Thread 2**: renderização do modelo 3D
- **Main Thread**: gerenciamento das mensagens do aplicativo

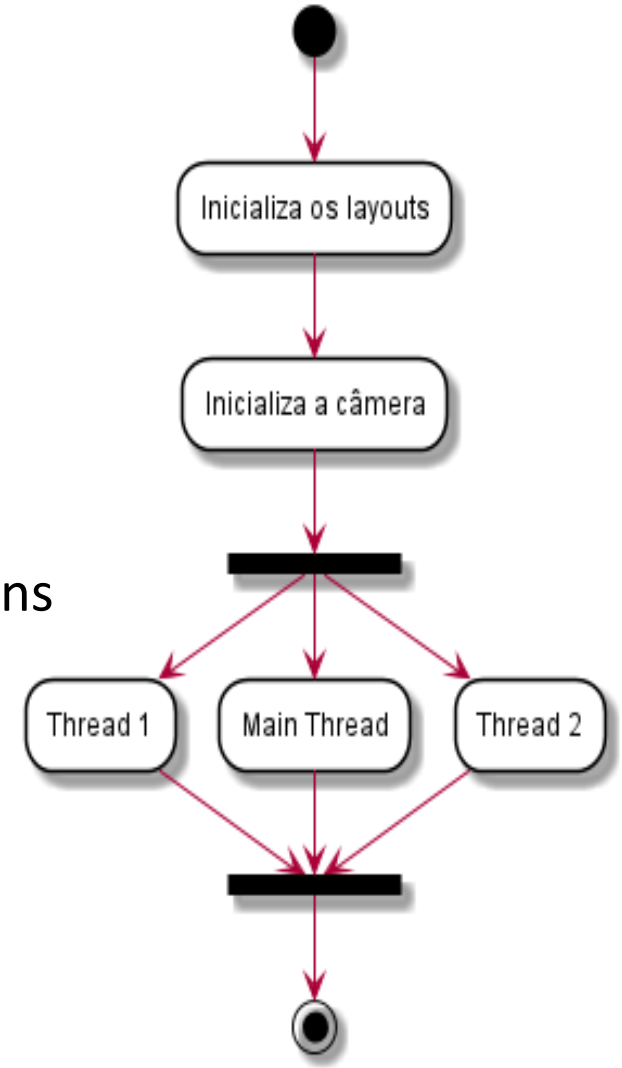
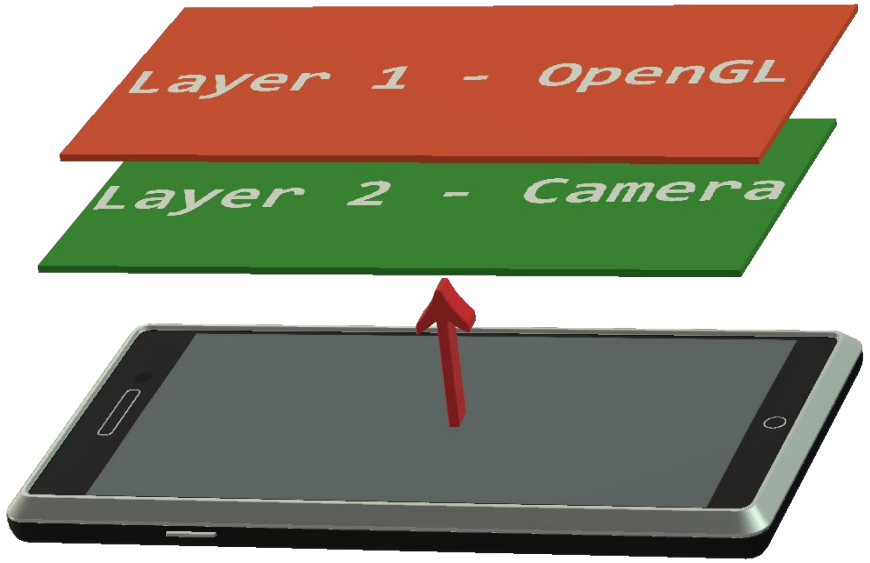


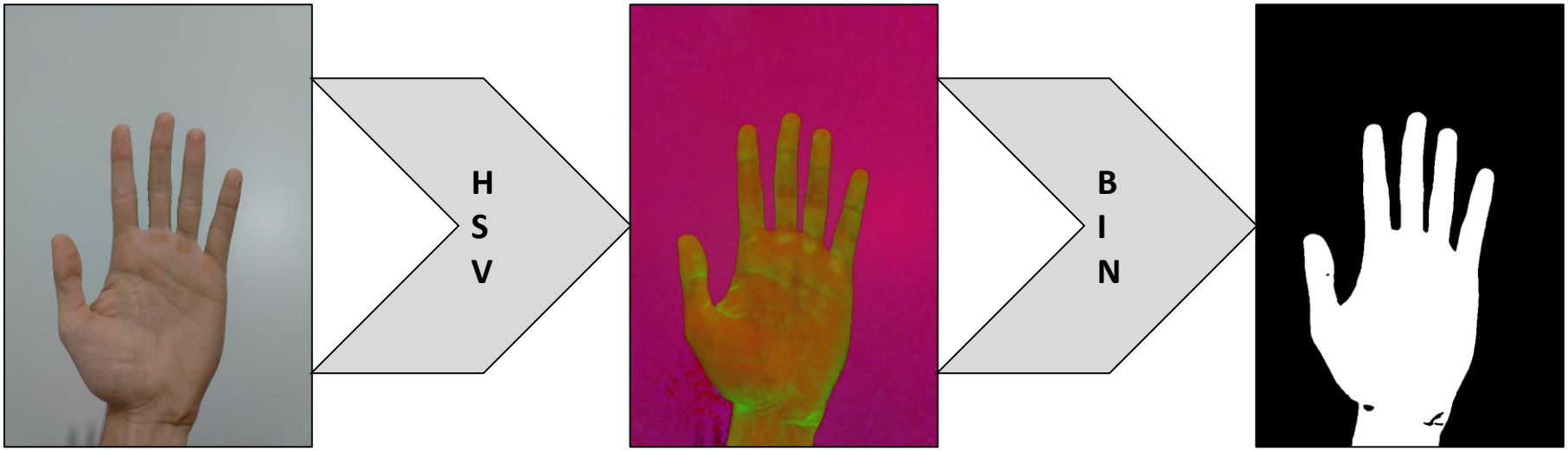
Diagrama de atividades

Layers do aplicativo

- Layer 1: desenhado pela *thread* do motor gráfico
- Layer 2: desenhado pela *thread* de identificação da mão



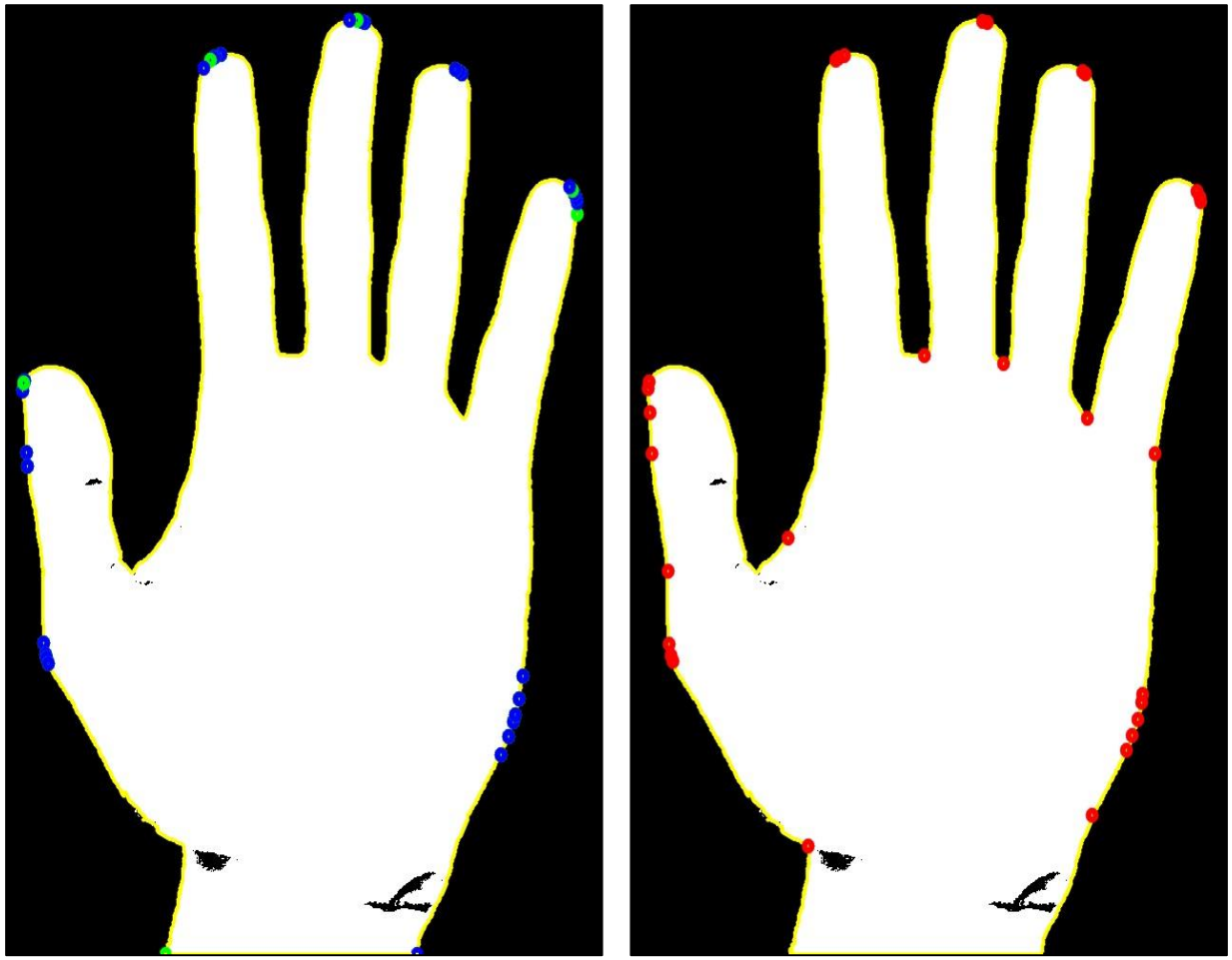
Conversão de imagem



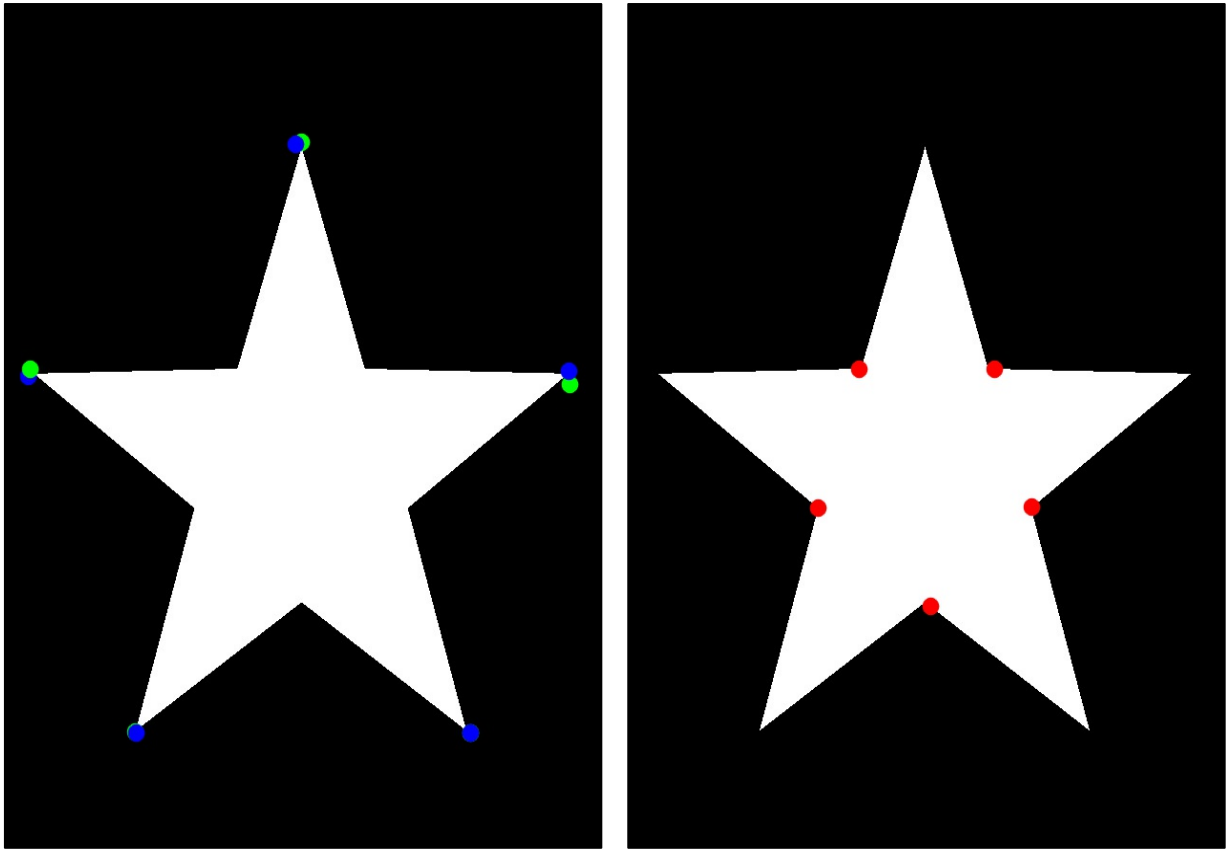
Matriz de defeitos de convexidade

Pontos convexos iniciais	Pontos convexos finais	Pontos de defeito de convexidade	Distância
24.00, 367.00	136.00, 718.00	198.00, 497.00	~ 126.24
137.00, 719.00	487.00, 719.00	316.00, 594.00	~125.00
487.00, 719.00	591.00, 375.00	422.00, 496.00	~126.75

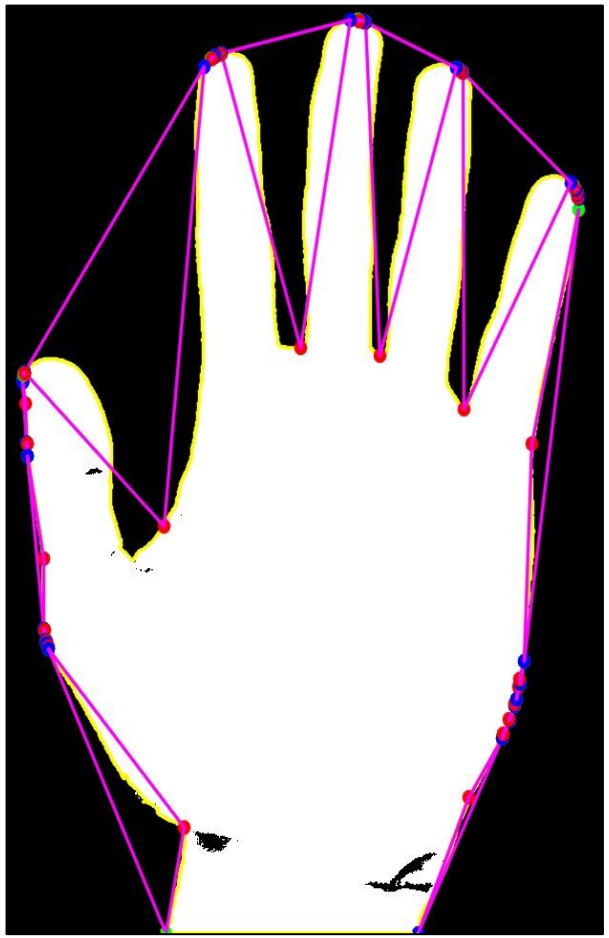
Pontos convexos da mão

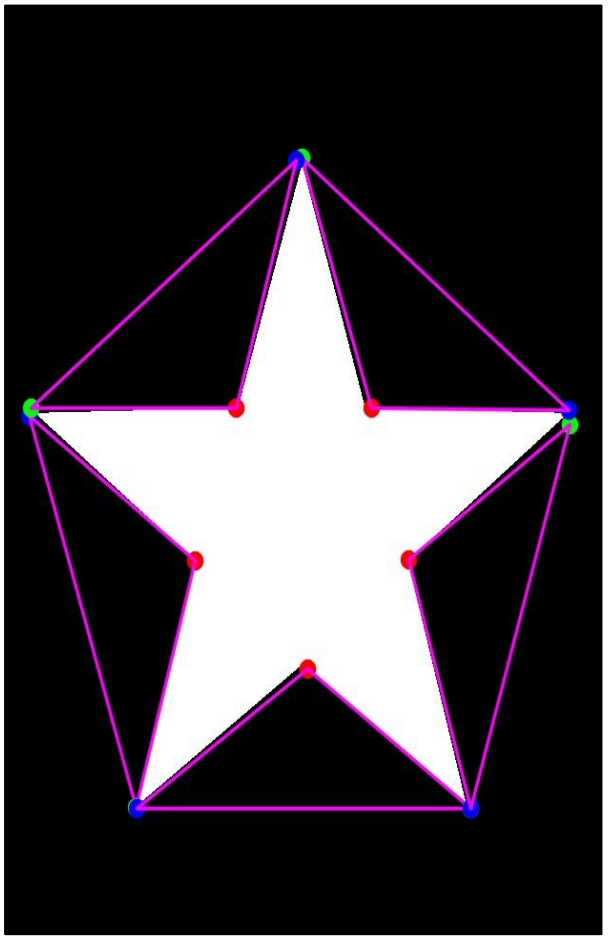


Pontos convexos

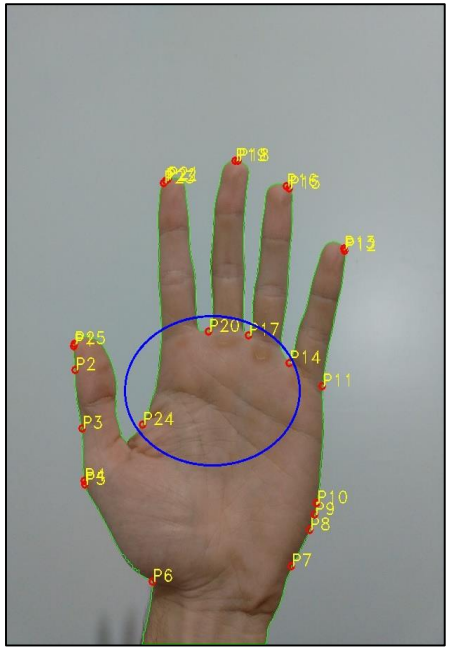


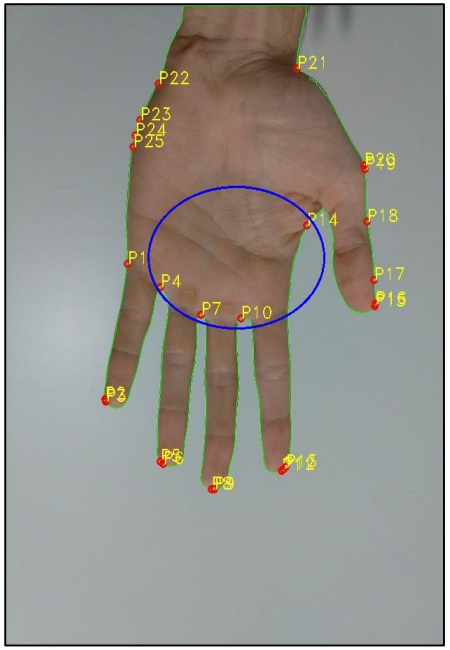
Triângulos de convexidade

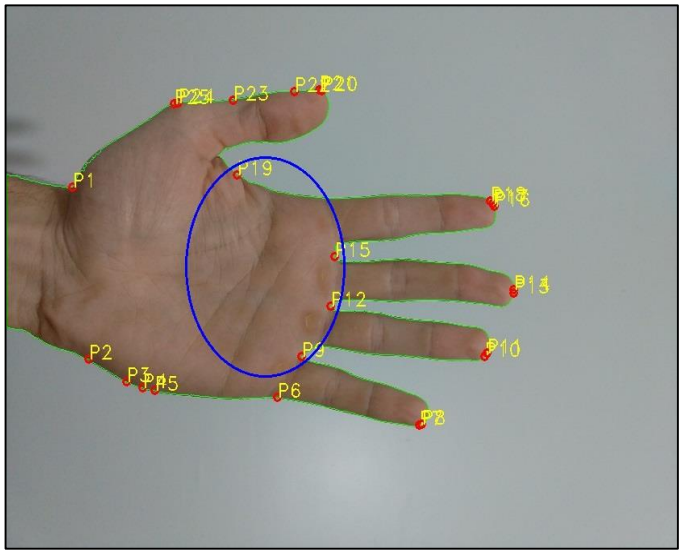


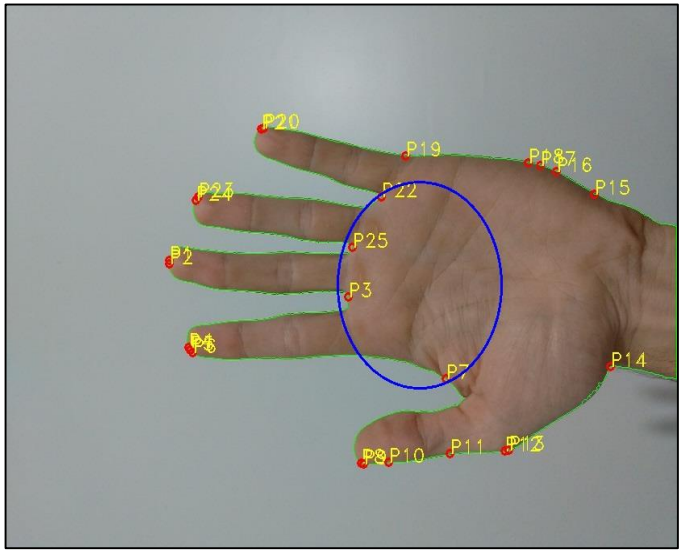


Ordenação dos pontos



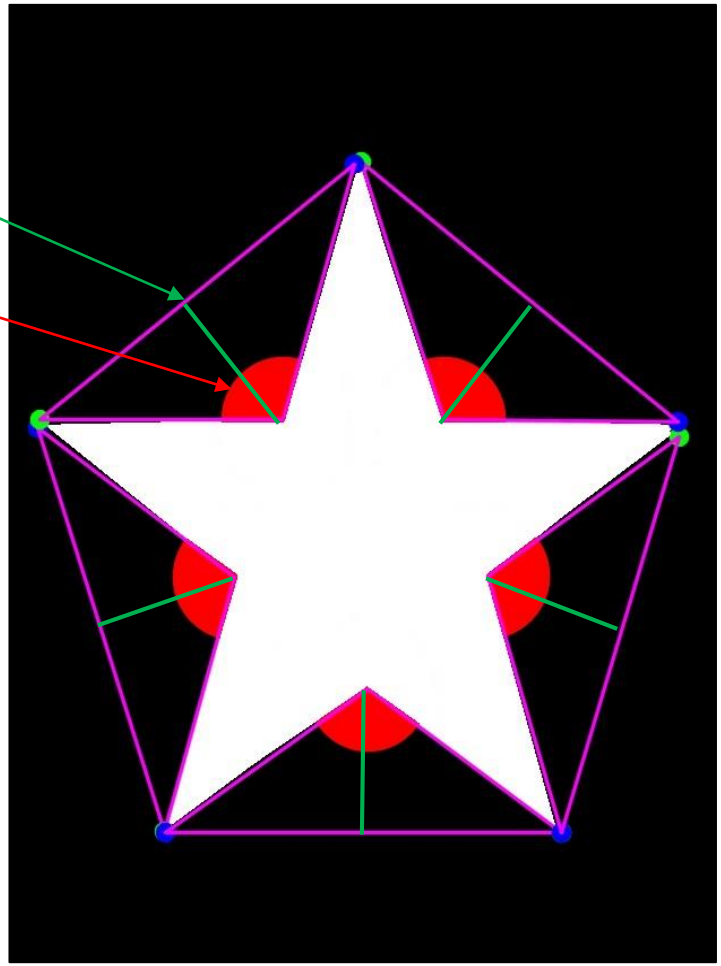




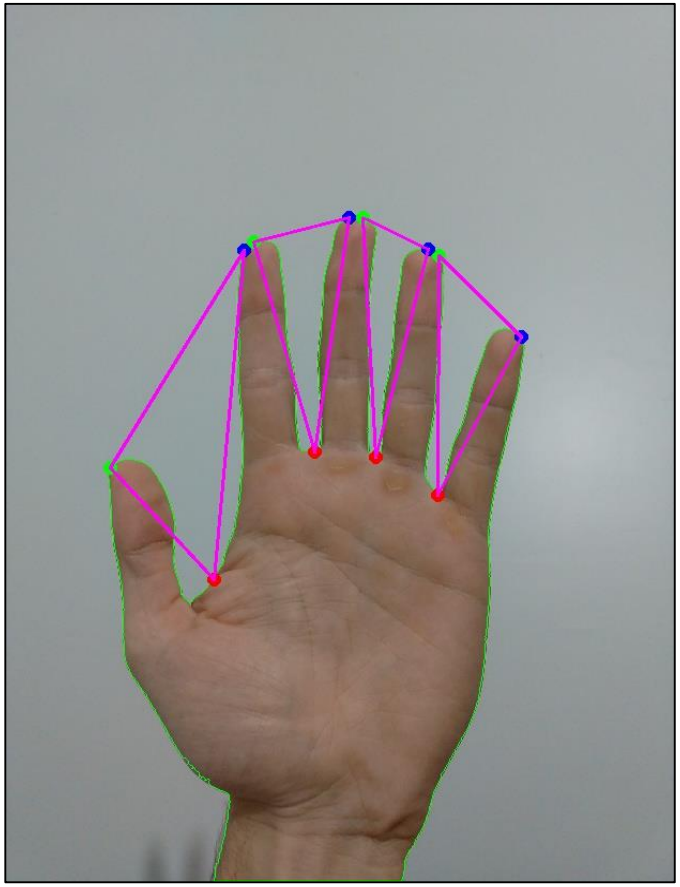


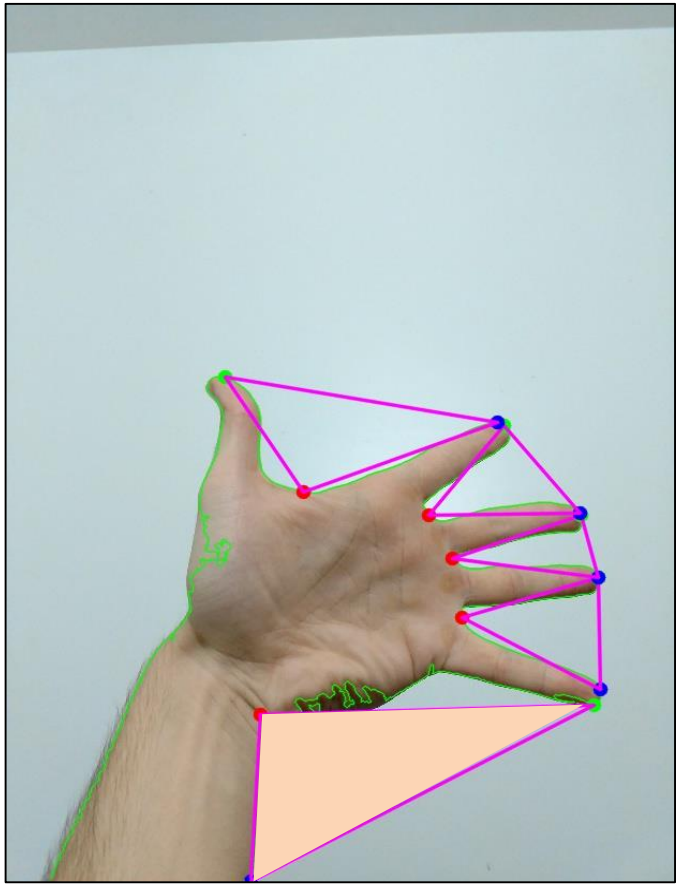
Filtrando os pontos convexos

- Altura do triângulo
- Ângulo interno



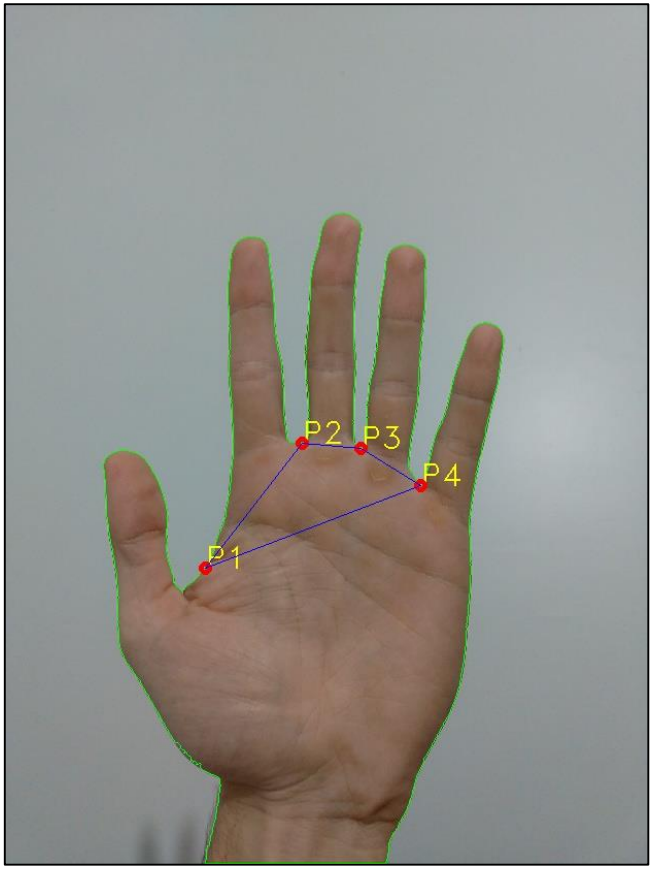
Triângulos de convexidade



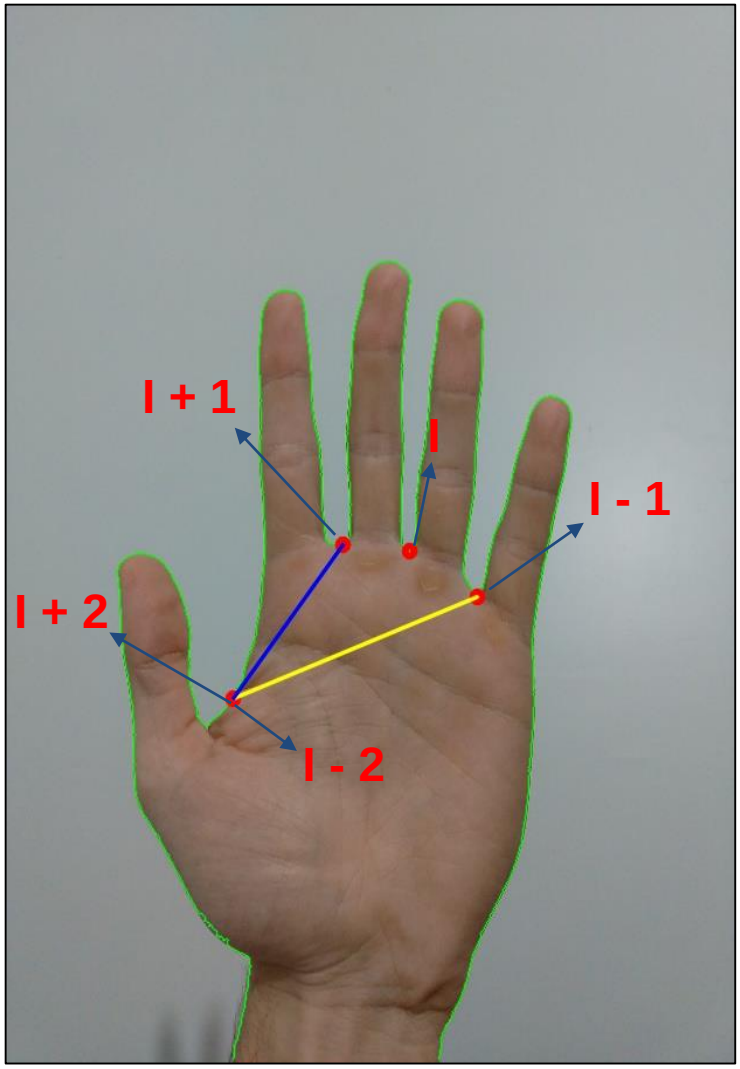


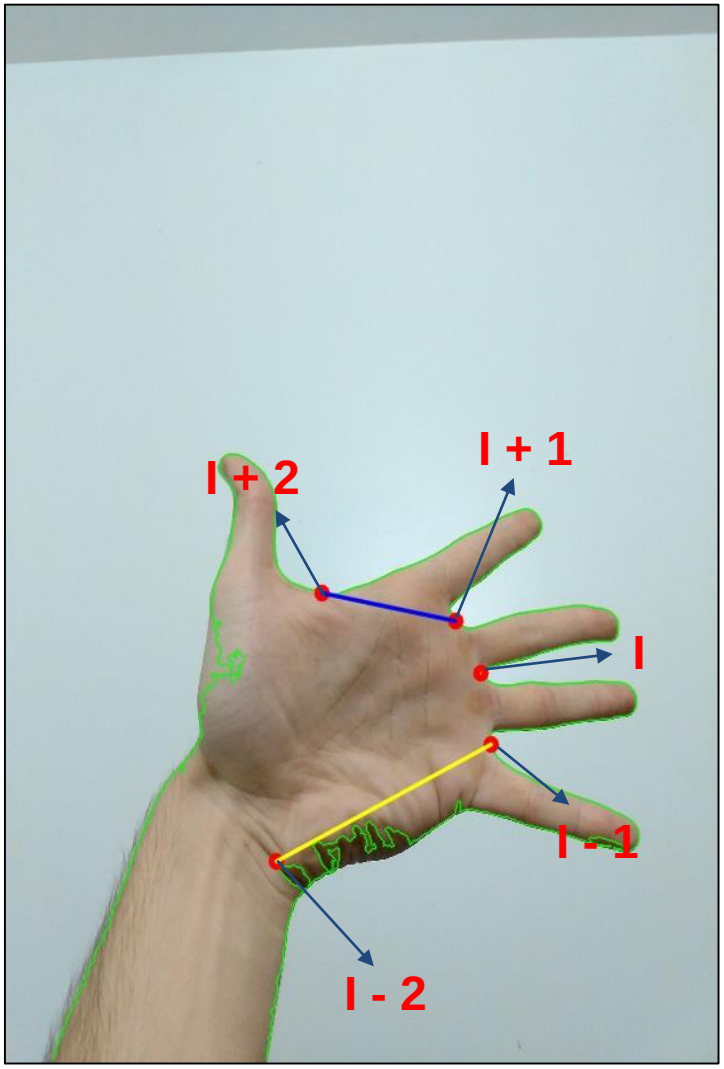
Ordenando a lista de pontos convexos

- `findClosest3Points()`
- retorna o índice do ponto pertencente ao dedo médio e o anelar



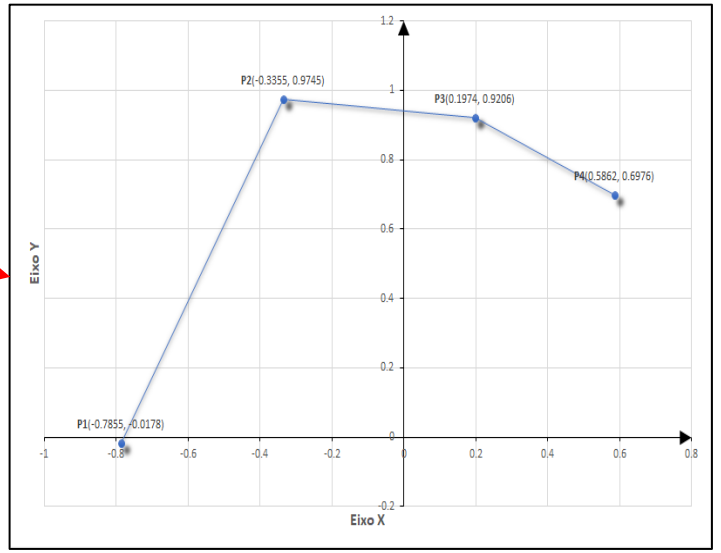
Localizando o polegar





Identificando a pose da mão

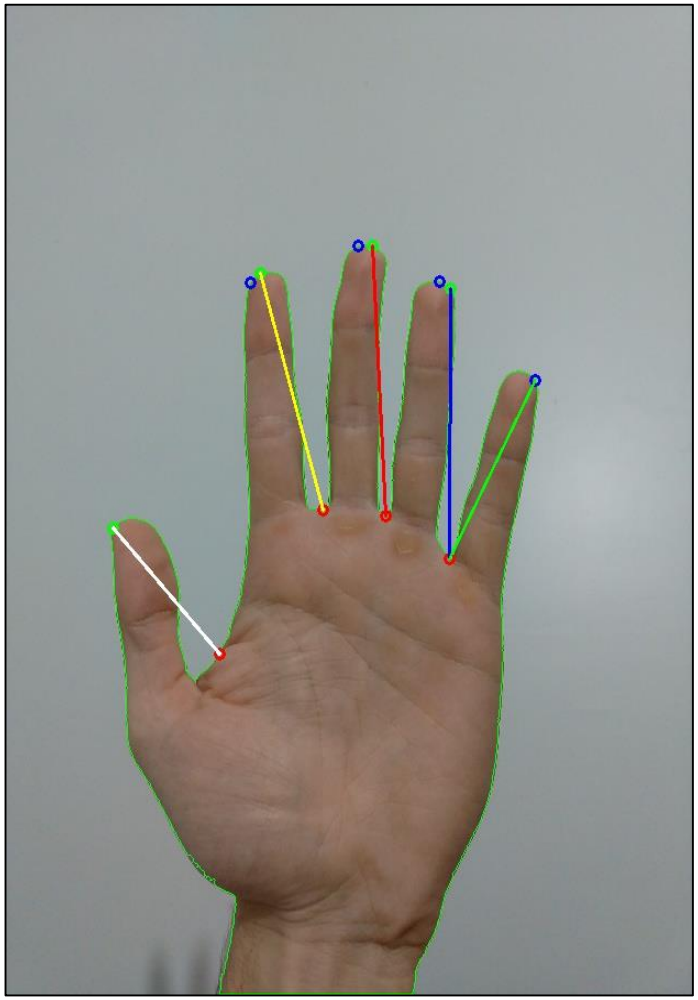
- solvePNP(A, B, C, D, ...)
- pontos de referência
- pontos da palma da mão
- parâmetros intrínsecos
- parâmetros extrínsecos



$$\begin{bmatrix} f_x & 0 & 0 \\ s & f_y & 0 \\ c_x & c_y & 1 \end{bmatrix}$$

Ângulo de rotação dos dedos

- Ângulo de rotação da palma: A
- Ângulo de rotação do dedo: B
- Ângulo de rotação padrão: C
- X = A - B - C



Atualizando os dados do objeto compartilhado

- Conteúdo da *flag* `render = true`
- Conteúdo da *flag* `right_model`
- Matriz 4x4 da pose da mão
- Lista de 5 posições com os ângulos dos dedos

Renderização do modelo 3D

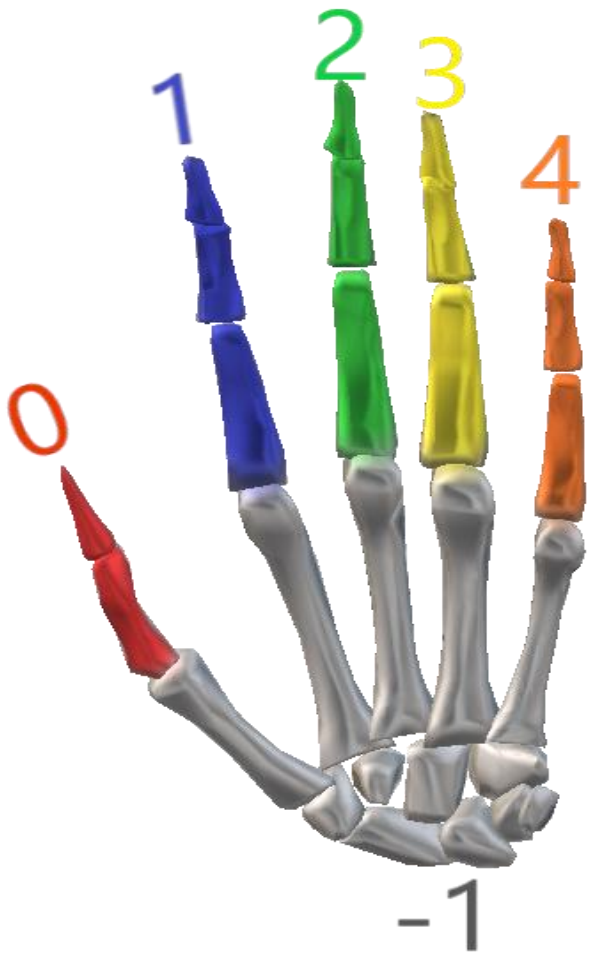
- Inicialização da cena
- Loop de renderização

Inicialização da cena

- Inicializa as matrizes temporárias
- Inicializa as matrizes *ModelView* de cada dedo
- Inicializa a matriz *ModelView* da palma
- Inicializa a luz da cena
- Carrega o modelo 3D
- Inicializa o contador mBadFrames

Identificando os objetos carregados

- Polegar
- Dedo Indicador
- Dedo Médio
- Dedo Anelar
- Dedo Mínimo
- Palma da mão e pulso



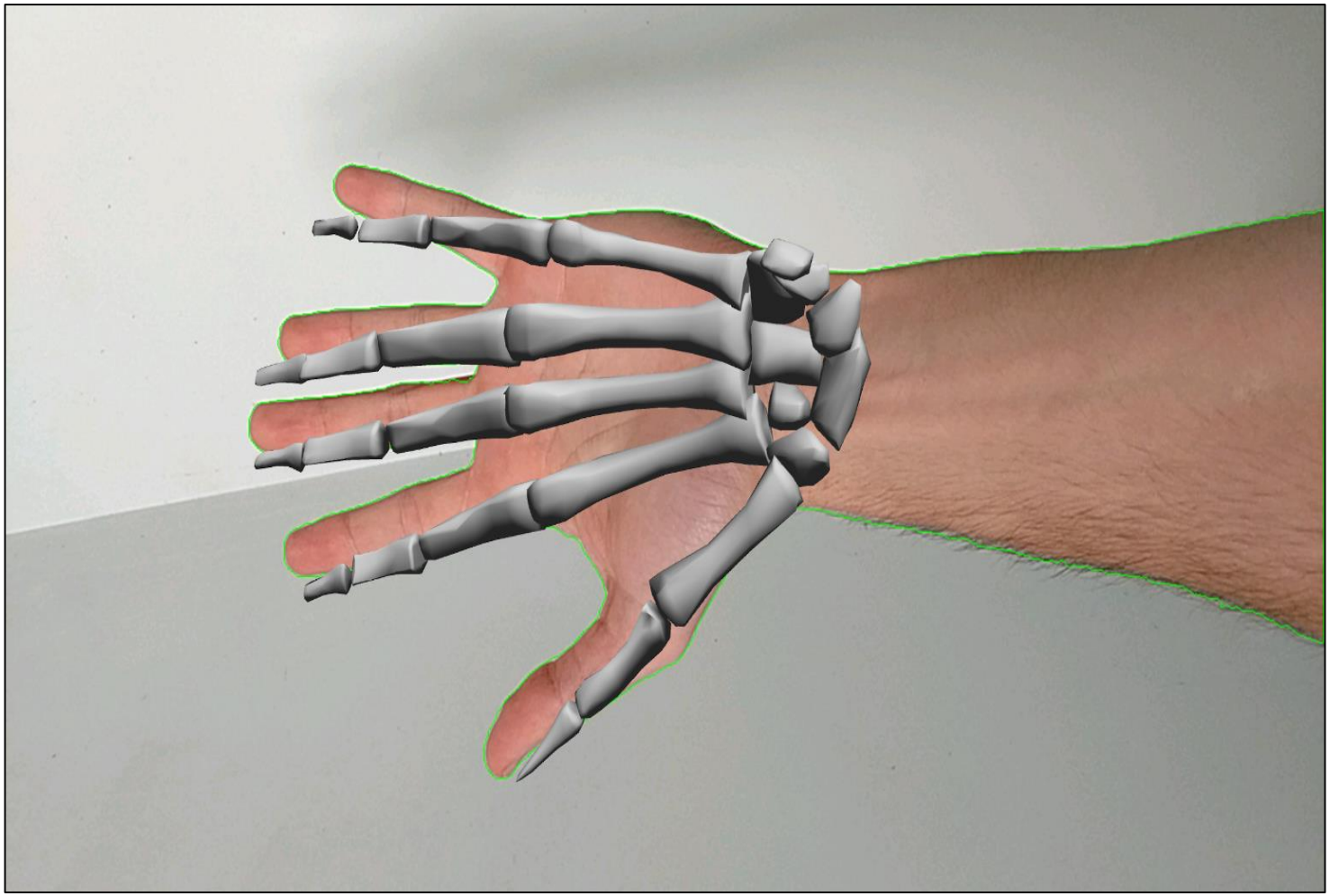
Loop de renderização

- Lê a informação do objeto compartilhado
- Compara **`mHandPose.render`** e **`mBadFrames < MAX_BAD_FRAMES`**
- Atualiza o valor do contador
- Muda a visibilidade do modelo 3D
- Atualiza a *ModelView* da mão com **`mHandPose.pose`**
- Calcula a ModelView de cada dedo
- Atualiza todos os objetos do modelo 3D com a nova *ModelView*

Calculando *ModelView* dos dedos

- Translada o dedo para a origem
- Rotaciona no eixo Z
- Translada de volta para posição original
- Aplica as transformações da *ModelView* da mão
- Gera a *ModelView* do dedo

Final do loop de renderização



Otimizações no módulo de câmera do OpenCV

- Novo parâmetro de câmera na inicialização:

  `setPreviewRange()`

- Nova alocação de imagem cache da câmera:

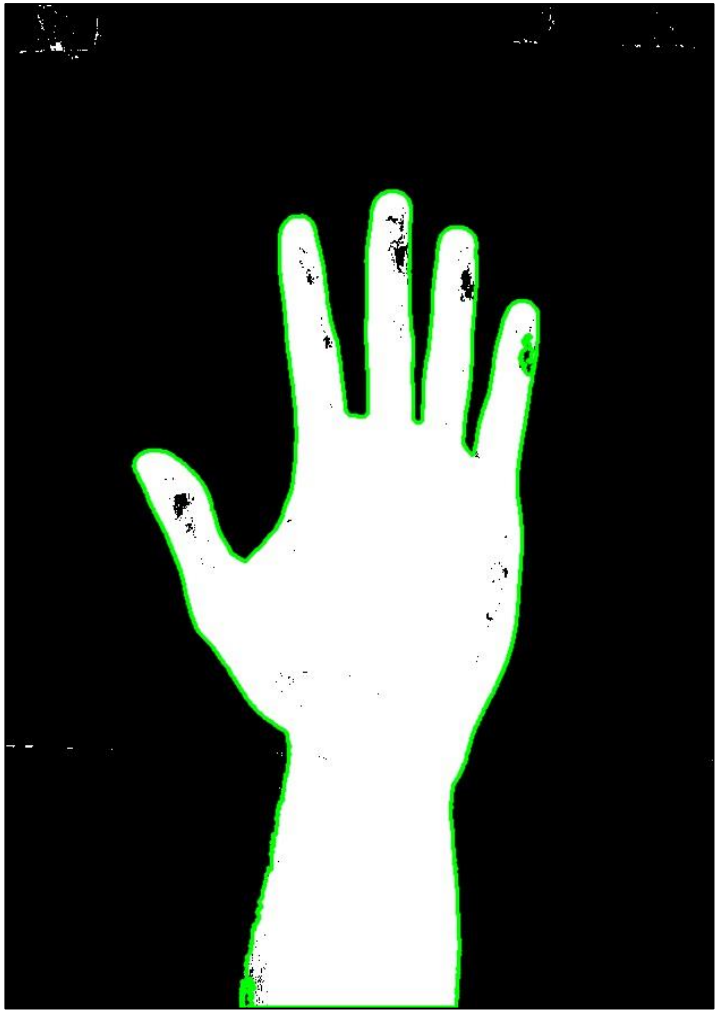
  `1280x720 / 2560x1440`

- Nova chamada da função de desenho
- Inicializações dentro do loop da *thread*

Resultados

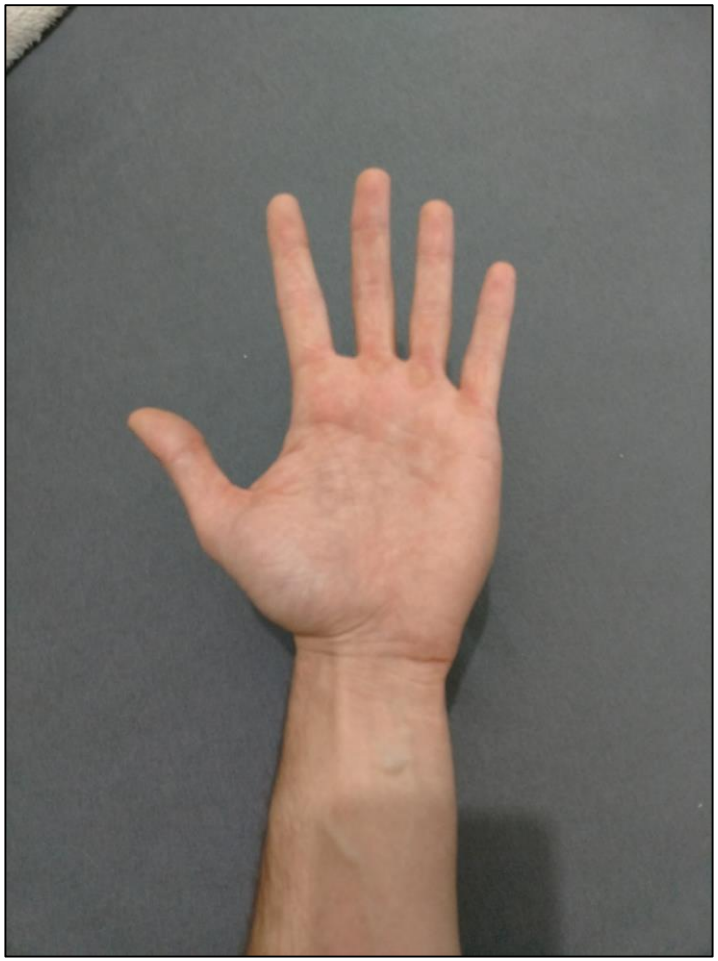
- Teste 01: Identificação do contorno
- Teste 02: Identificação dos principais pontos convexos
- Teste 03: Renderização do modelo 3D com a pose da mão
- Teste 04: Otimização do módulo de câmera do OpenCV

Teste 01: Identificação do contorno



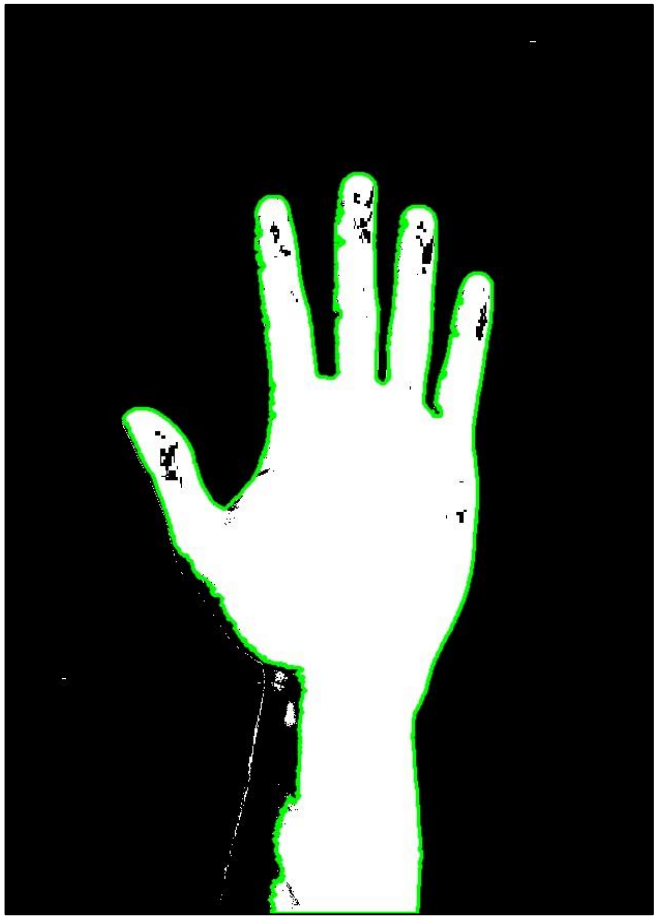
Ambiente controlado com boa iluminação

Teste 01: Identificação do contorno



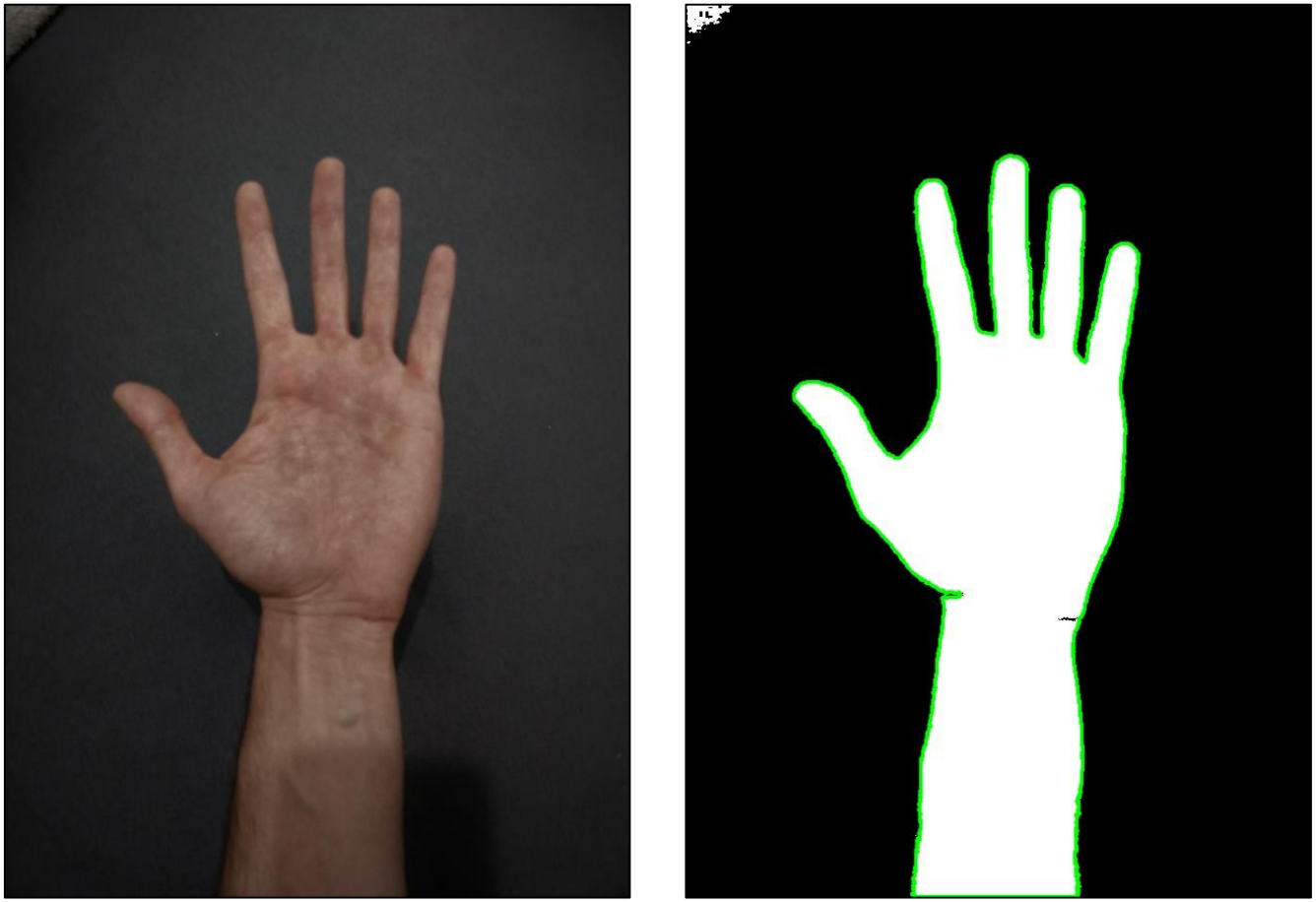
Ambiente controlado com boa iluminação

Teste 01: Identificação do contorno



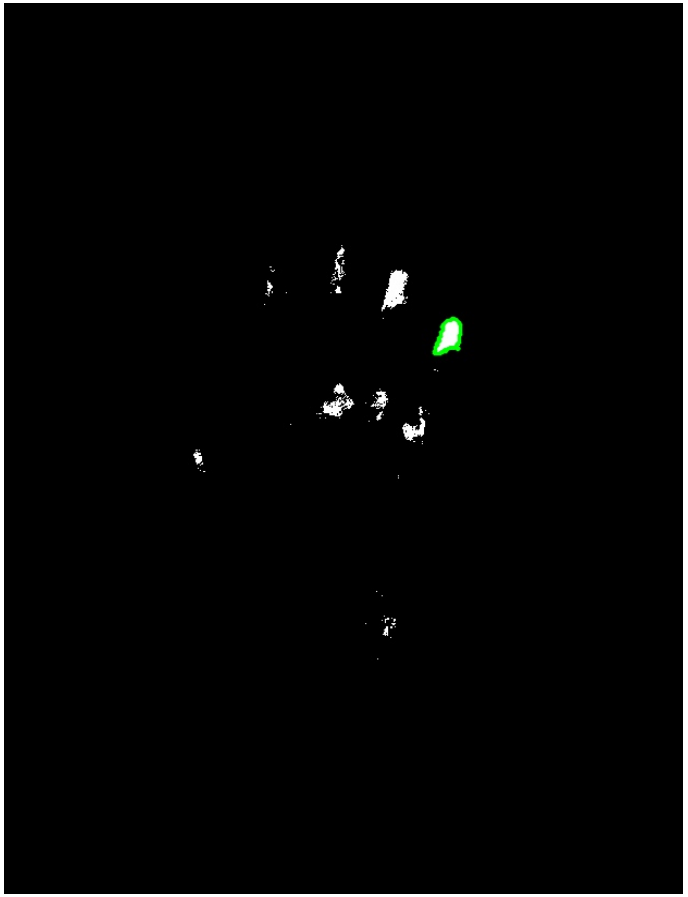
Ambiente controlado com iluminação moderada

Teste 01: Identificação do contorno



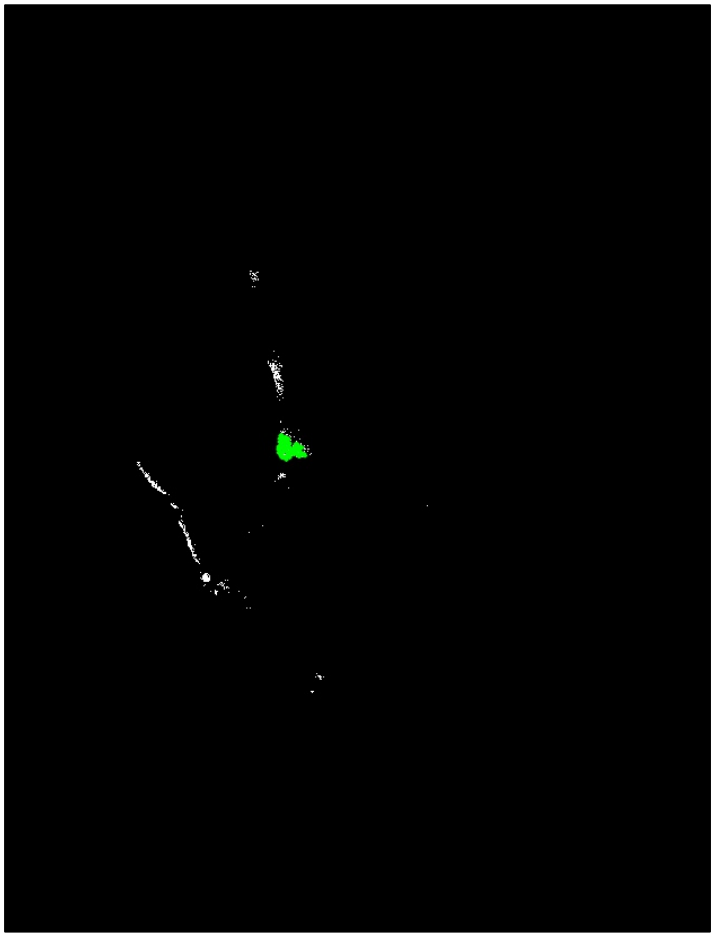
Ambiente controlado com iluminação moderada

Teste 01: Identificação do contorno



Ambiente controlado com baixa iluminação

Teste 01: Identificação do contorno

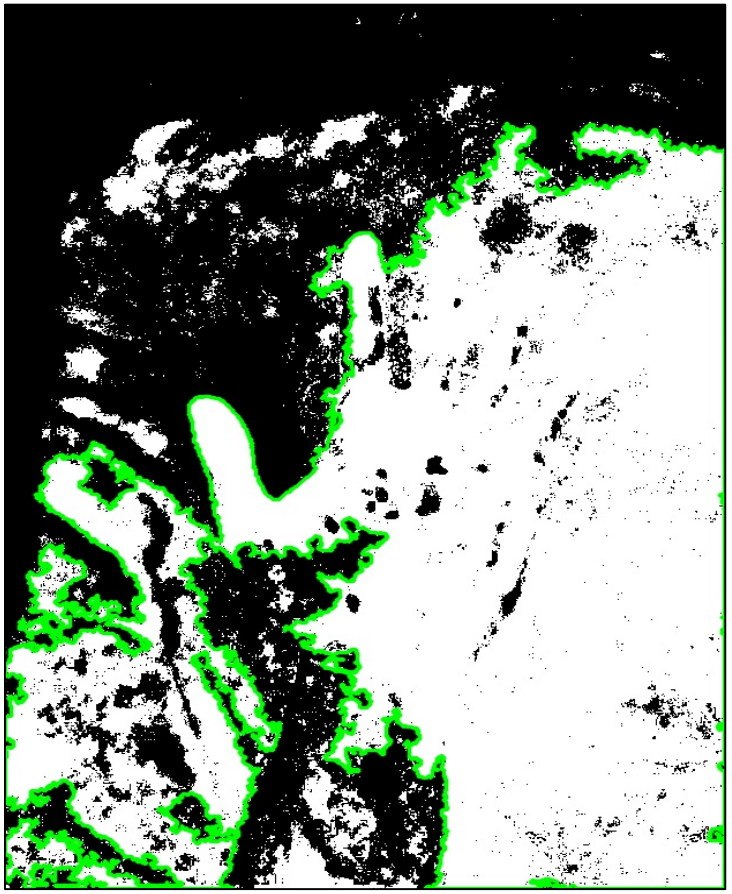


Ambiente controlado com baixa iluminação

Teste 01: Identificação do contorno

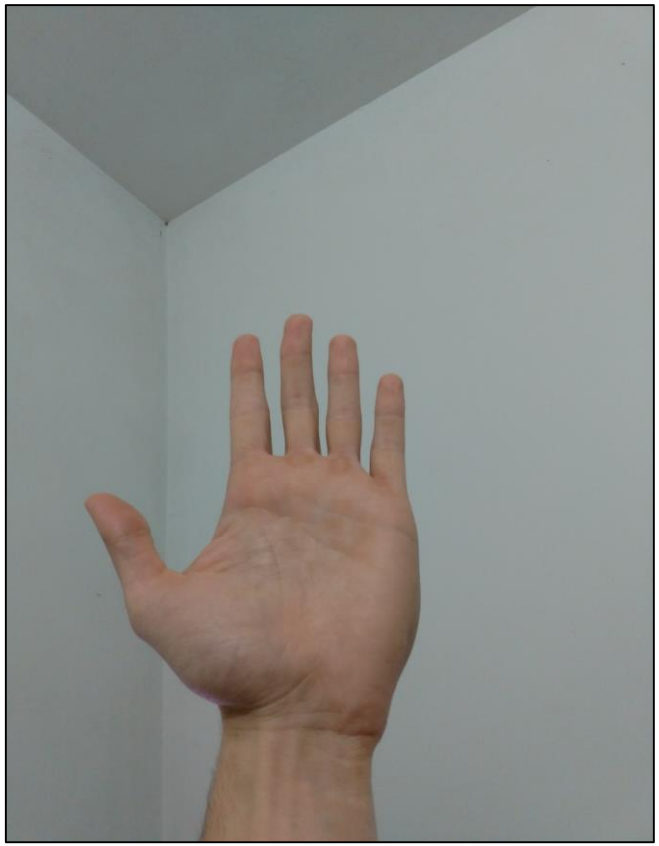
Ambiente diverso com boa iluminação

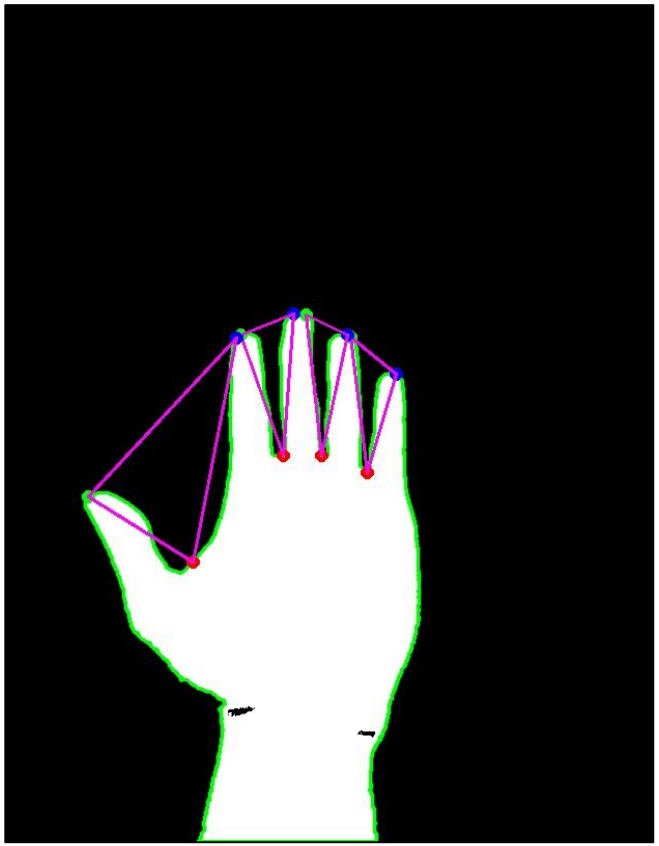
Teste 01: Identificação do contorno



Ambiente diverso com boa iluminação

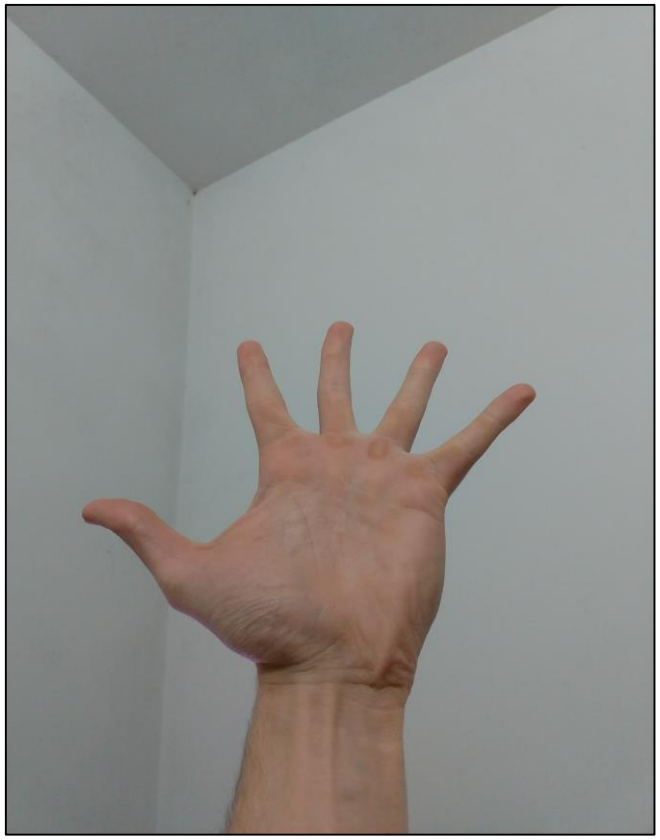
Teste 02: Identificação dos principais pontos convexos

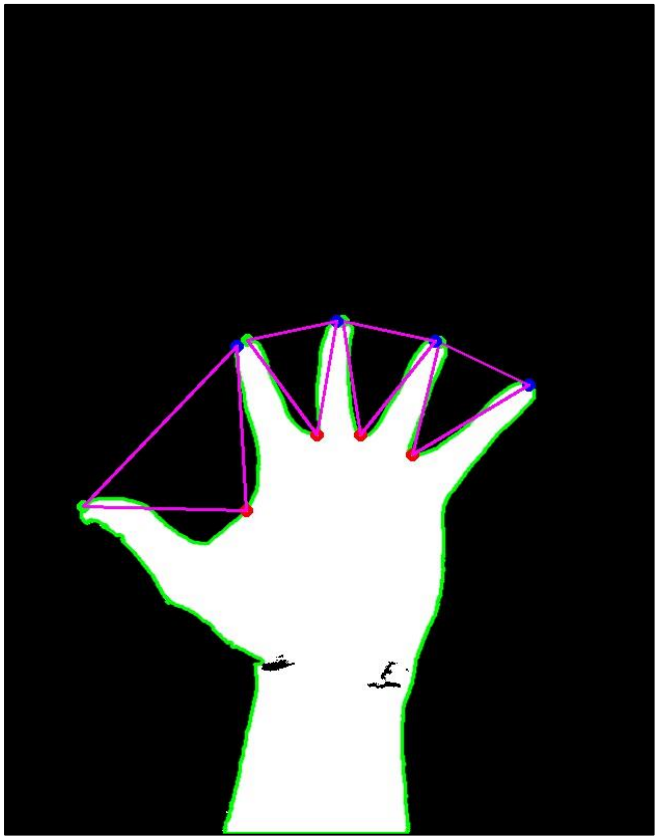




Dedos próximos

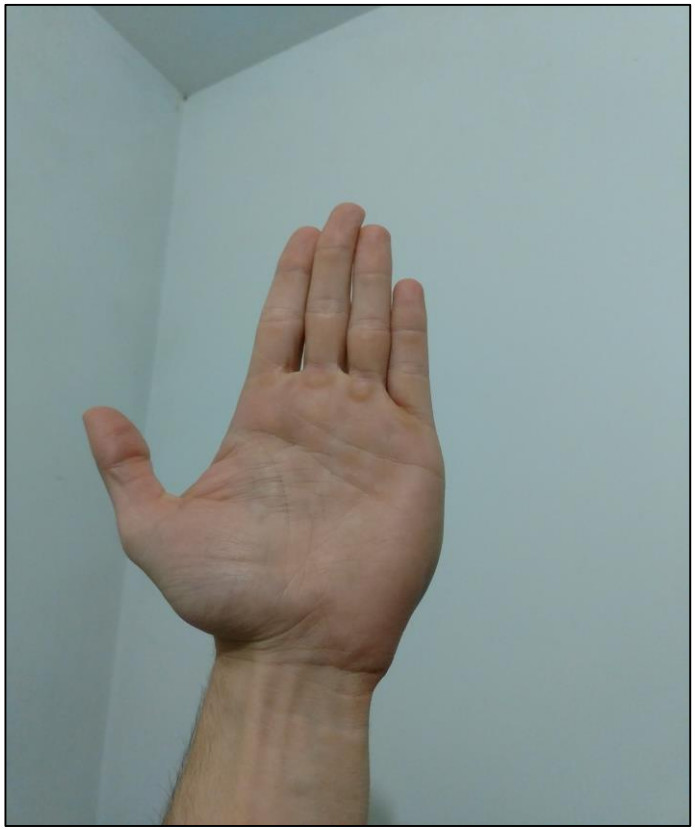
Teste 02: Identificação dos principais pontos convexos

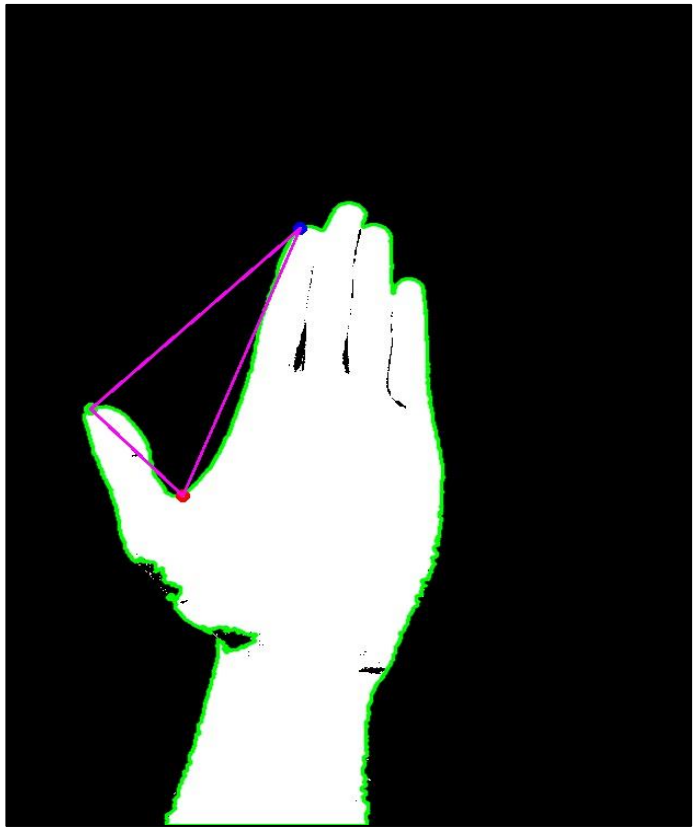




Dedos bem afastados

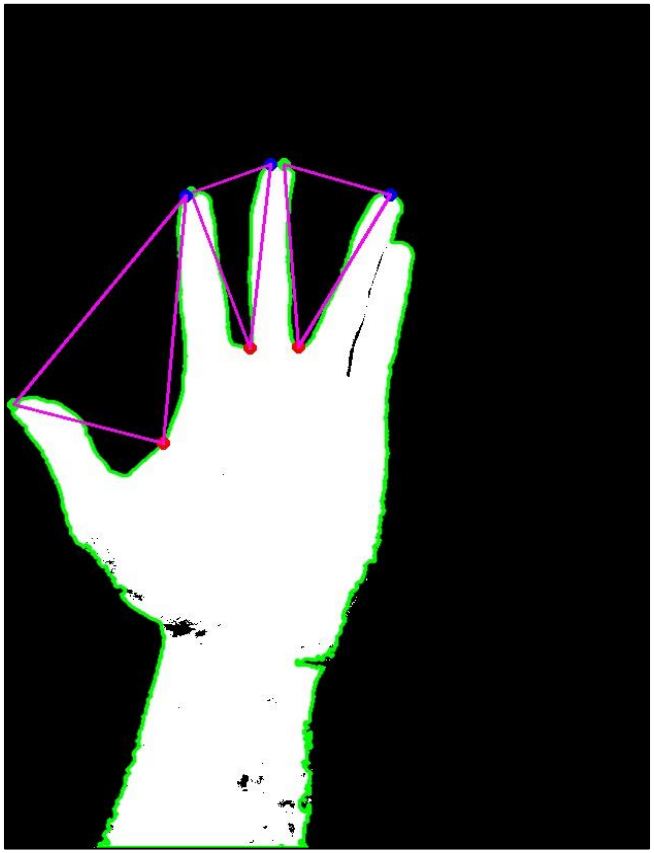
Teste 02: Identificação dos principais pontos convexos





Dedos muito agrupados

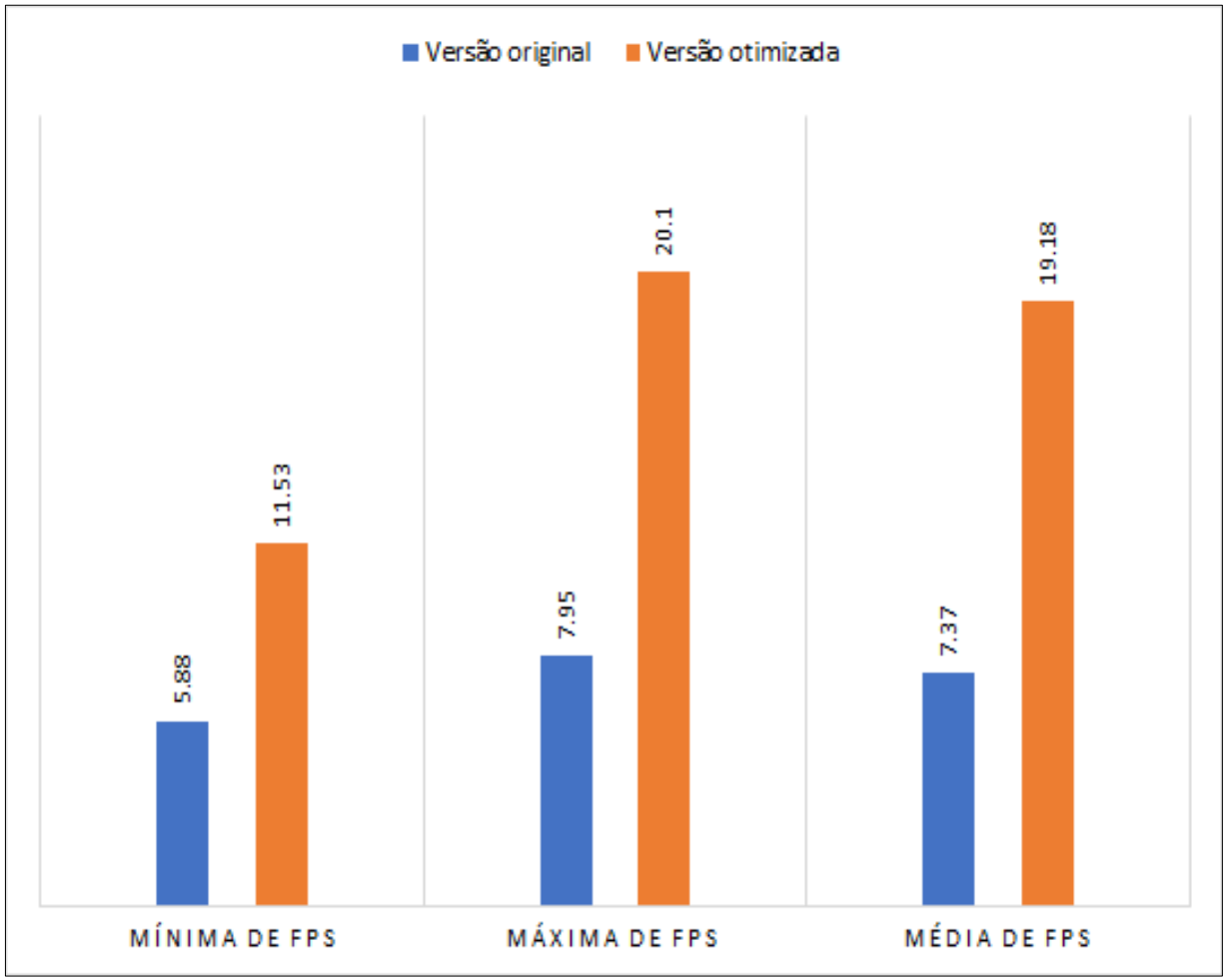
Teste 02: Identificação dos principais pontos convexos



Dedos muito agrupados

Teste 03: Renderização do modelo 3D com a pose da mão

Teste 04: Otimização do módulo de câmera do OpenCV



Conclusões

- Identificação do contorno da mão:
  - Ambientes controlados com boa iluminação
  - Ambientes controlados com iluminação moderada
- Identificação dos principais pontos convexos:
  - Dedos separados
- Renderização do modelo 3D
  - Pequeno desalinhamento do modelo 3D
- Otimização do módulo de câmera do OpenCV
  - Ganho de desempenho em ~160%

Extensões

1. melhorar ou utilizar outras técnicas de segmentação para encontrar a mão
2. estimar a pose da mão através de técnicas de inteligência artificial como Redes Neurais
3. implementar a visualização de todos os sistemas fisiológicos
4. transformar em uma aplicação de realidade virtual

Obrigado!