Sistemas de Visão e Percepção Industrial

Aula Prática nº 13

Exercícios em Sherlock - Parte 2

Universidade de Aveiro
2021-2022

Sumário

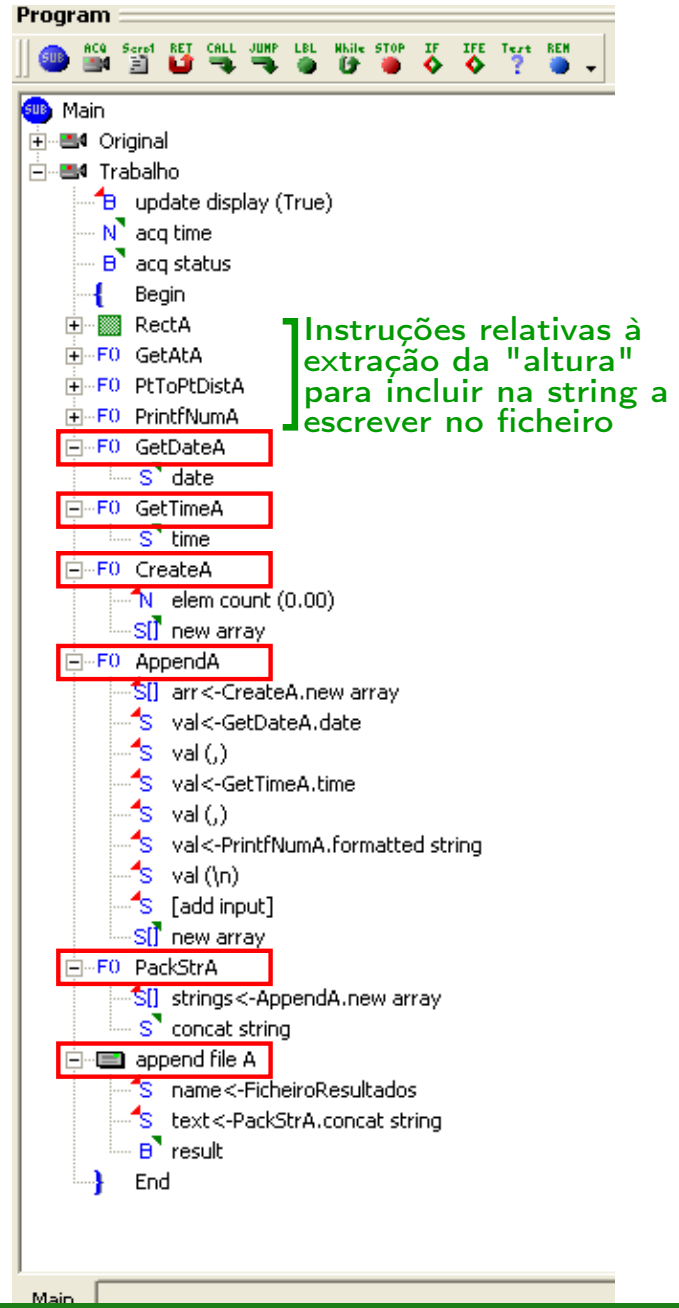
Exercício 1 – Escrita em ficheiro

- Complementar e adaptar o exercício 5 da aula anterior de forma a escrever os dados num ficheiro de nome: "resultados.txt"
  - Sugestão: criar uma variável com o caminho exato para o ficheiro de resultados para se garantir que escreve no local correto.
- Cada imagem deve gerar dados para uma linha que deve conter a seguinte informação:
  - (Data, hora, medida) separados por vírgulas e terminados com caracteres de mudança de linha ("\n")
    - 05/06/22, 10:51:16, 113
    - 05/06/22, 10:51:22, 56
    - 05/06/22, 10:51:23, 113
    - 05/06/22, 10:51:24, 179
    - Etc...

Exercício 1 – Algumas funções úteis

- IO: System:GetDate
  - Data do sistema
- IO: System:GetTime
  - Hora do sistema
- Array: String:Create
  - Criação de um array (vazio) de strings
- Array: String:Append
  - Adição de string(s) a um array
- String: PackStr
  - Criação de uma única string por "enpacotamento" de array
- IO: File:Append
  - Acrescentar string a um ficheiro
  - Cria o ficheiro se não existir

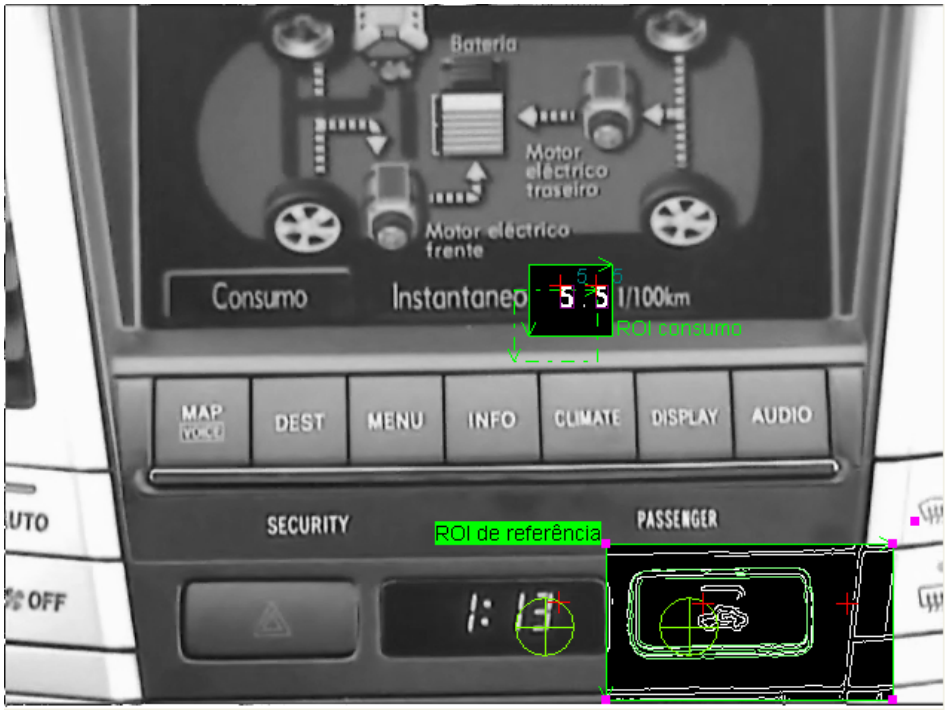
N.B. Há muitas outras funções e formas de lidar com strings e arrays de strings.



Exercício 2 – Alinhamentos

- Criar uma aplicação em Sherlock para fazer o alinhamento automático da imagem para permitir o seu posicionamento automático e assim vir a detetar e analisar regiões específicas de interesse.
  - Usar a sequência das imagens 0 a 23
- Procurar features (edges, ou patterns) que possam ajudar a "fixar" a imagem

Exercício 2 – Passos e sugestões



1. Definir um modelo (fração de imagem ou "pattern" na linguagem do Sherlock) a procurar por "search geometric"
2. Editá-lo em conformidade (remover linhas e/ou pontos)
   - Criam-se elementos para definir alinhamentos (neste caso: best point, pattern left, pattern right)
3. Criar uma ROI e associá-la ao alinhamento anterior.
4. A nova ROI seguirá sempre a referência!

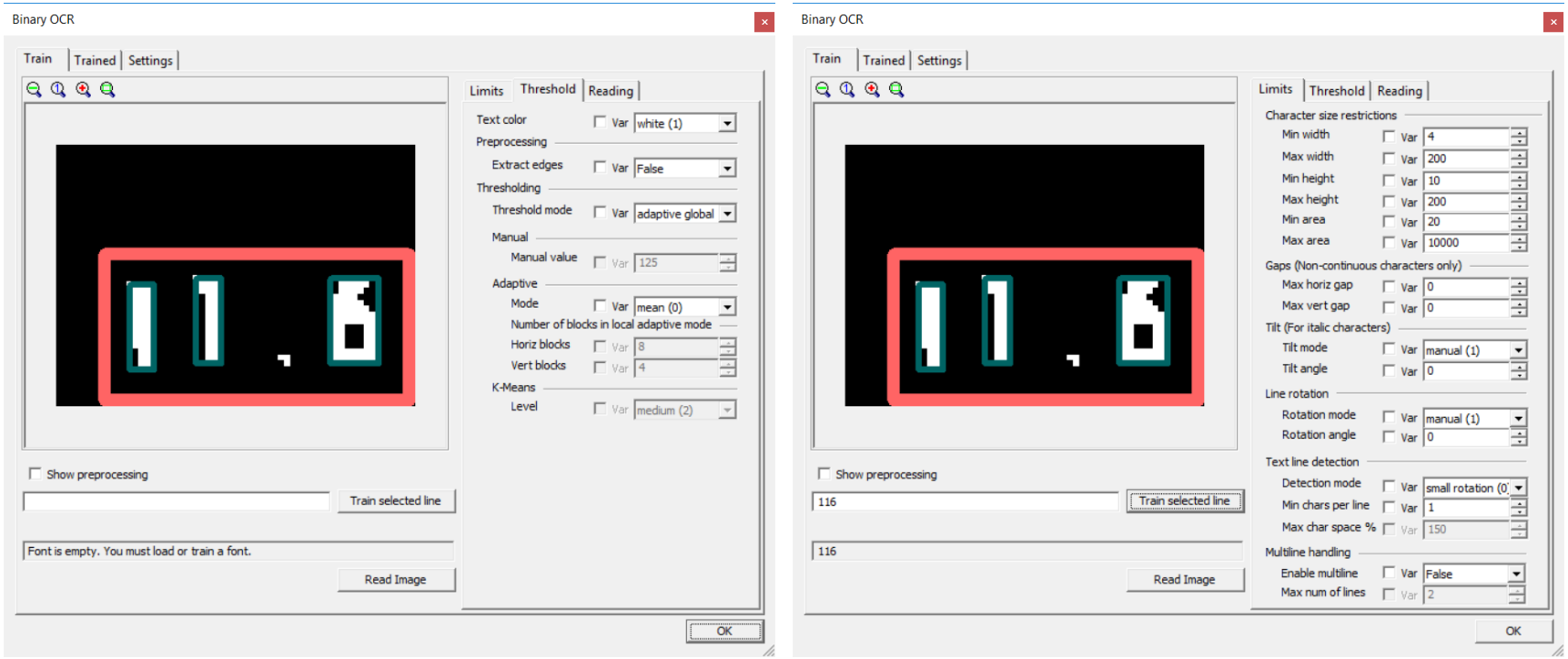
Exercício 3 – OCR (binary OCR)

- Criar uma aplicação em Sherlock para fazer o reconhecimento automático de caracteres usando binary OCR e depois guardar o texto obtido num ficheiro de resultados.
- A sequência de imagens vem do mostrador de um carro onde se vêm diversas informações, sendo a que interessa a do consumo instantâneo.

Valor que interessa extrair.

Exercício 3 – Sugestões e exemplos

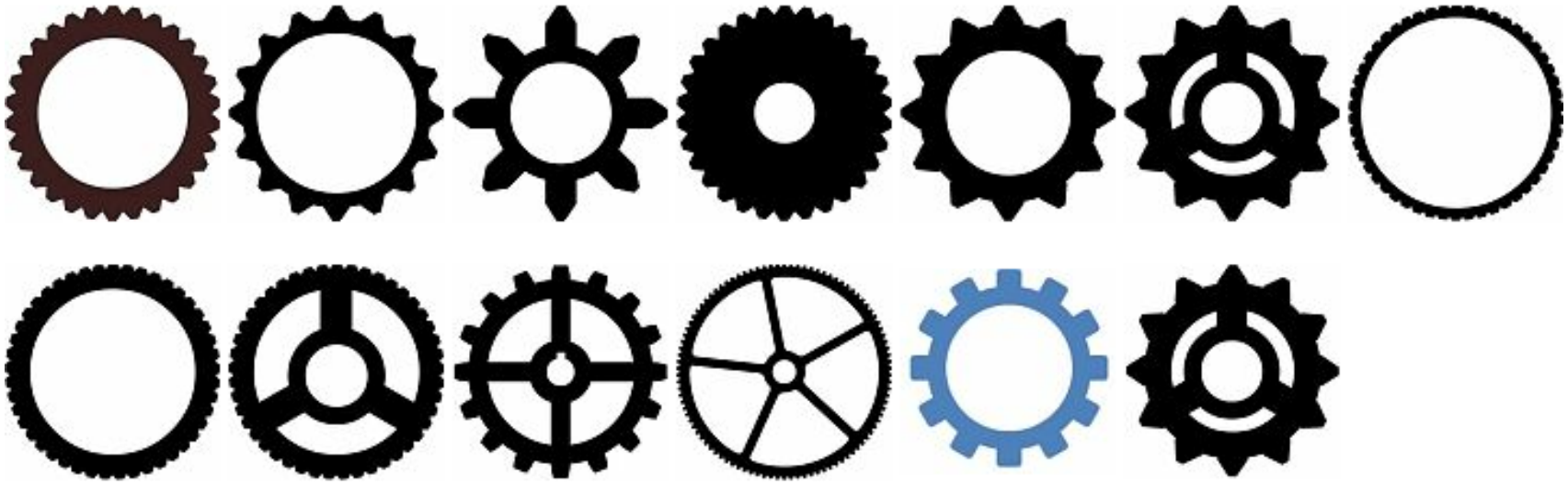
- O uso de binary OCR deve ser configurado e "treinado" para aprender a ler carateres, tendo em atenção se os caracteres são em branco ou preto, bem como as suas dimensões mínimas:



- O treino pode ser feito na sequência das imagens 0 a 23
- Mas depois testado nas outras sequências (31-62 e/ou 69-80)

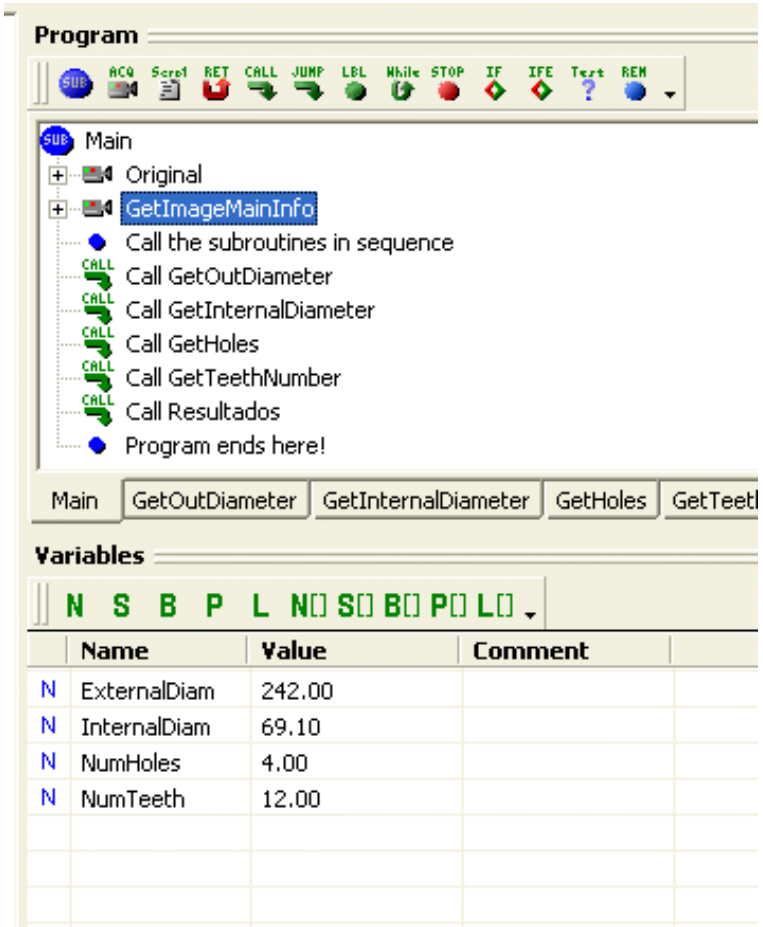
Exercício 4 – Propriedades de rodas dentadas

- Fazer um programa em Sherlock para determinar propriedades das rodas dentadas da sequência "gear_xxxx.png"
  - Diâmetros interno e externo, número de furos e de dentes.
- Organizar o programa em subrotinas, usando uma subrotina para calcular cada propriedade, que serão chamadas do "main".



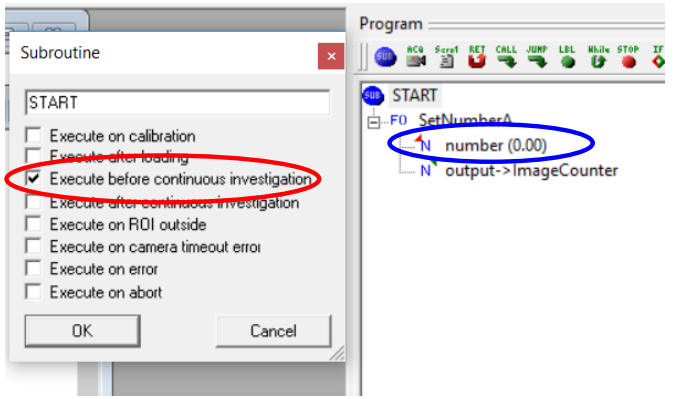
Exercício 4a) – Programa Geral

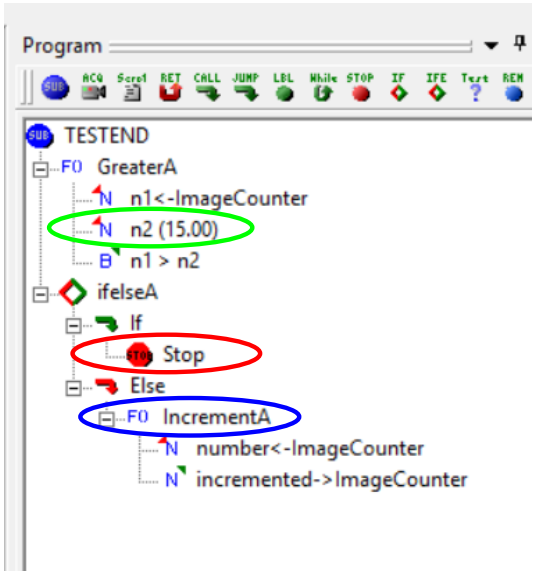
- Carregar sequência de imagens
- Criar a ROI inicial
- Calcular dimensões extremas do objeto
- Calcular reposicionamento para ROIs
- Criar variáveis para as grandezas a calcular:
  - Diâmetro externo
  - Diâmetro interno
  - Número de furos
  - Número de dentes
- Criar o espaço (vazio) para as diversas subrotinas.
- Criar as chamadas das subrotinas com "Call"



Nota sobre subrotinas

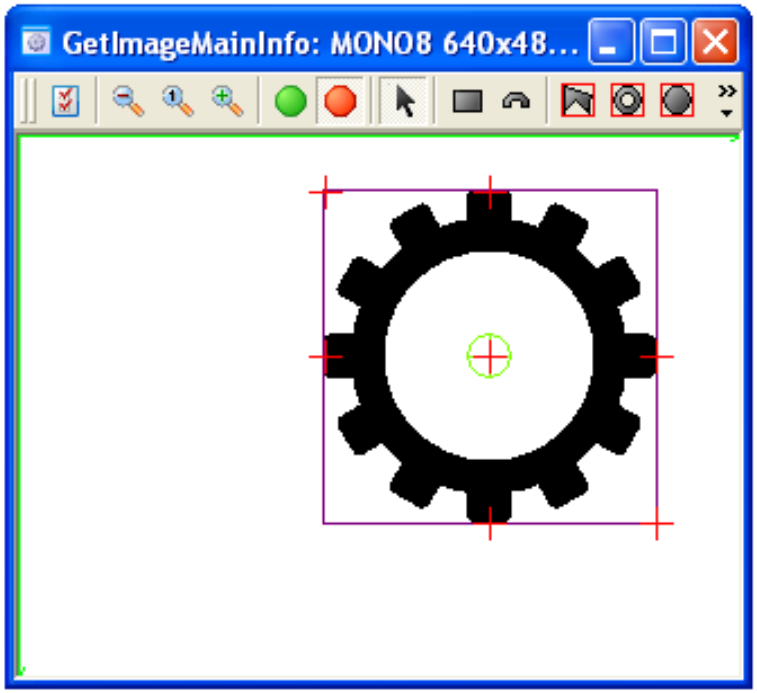
- É possível criar uma subrotina (designada START no exemplo) que executa **apenas uma vez** e antes da execução contínua.
  - Isso pode ser usado para inicializar variáveis ou outras ações únicas.
- Por outro lado, pode-se controlar certas atividades antes de carregar cada nova imagem para tomar eventuais decisões. Por exemplo, parar a execução contínua se se tiver atingido uma certa condição.
  - No exemplo ilustrado, termina-se a execução se a variável ImageCounter superar o valor 15, e incrementa-se esse contador em caso contrário.
- Esta subrotina TESTEND deverá ser executada no início do programa principal.

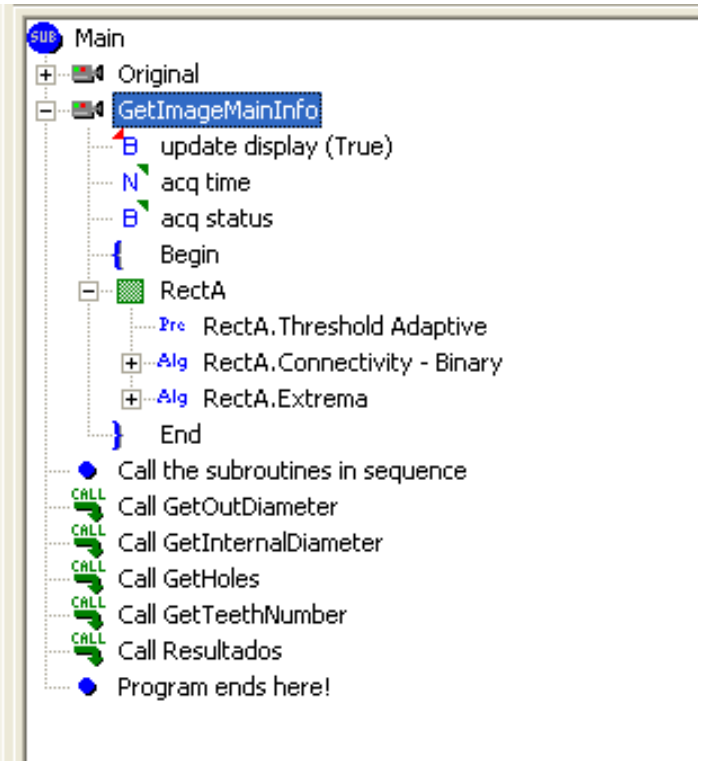




Exercício 4b) – Programa principal

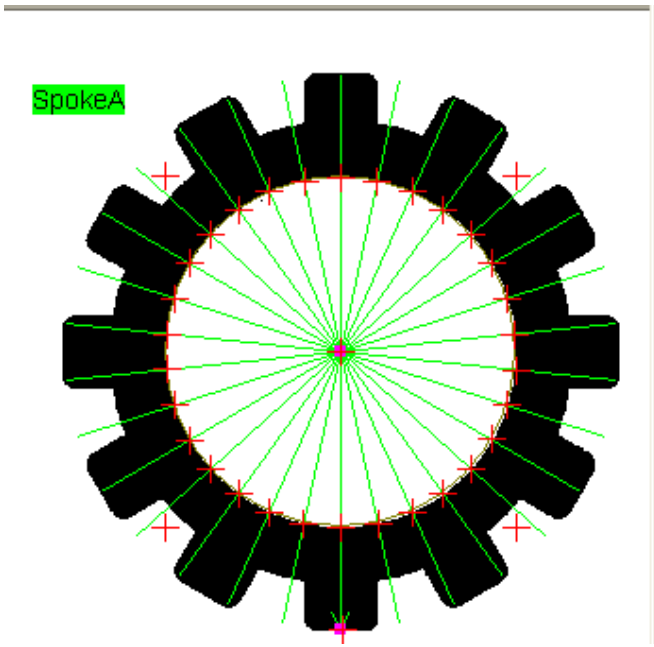
- Definir a ROI global.
- Usar Connectivity binary – para detetar o blob e o seu centro
- Obter os extremos da região binarizada com "extrema" (Algorithm)
- Escrever rotina GetOutDiameter para calcular diâmetro externo em função dos "extrema"

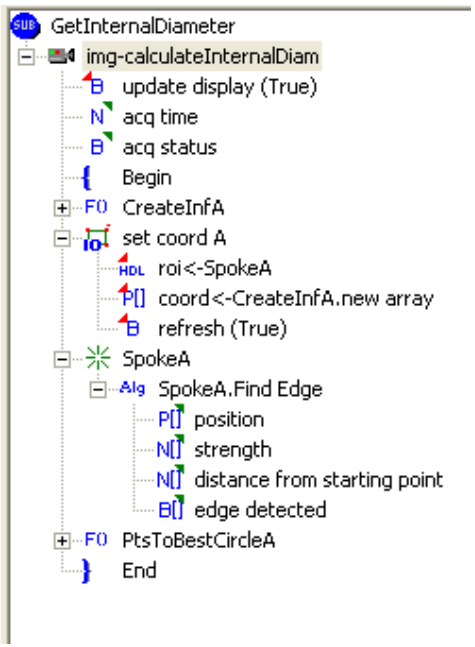




Exercício 5) – Diâmetro interno

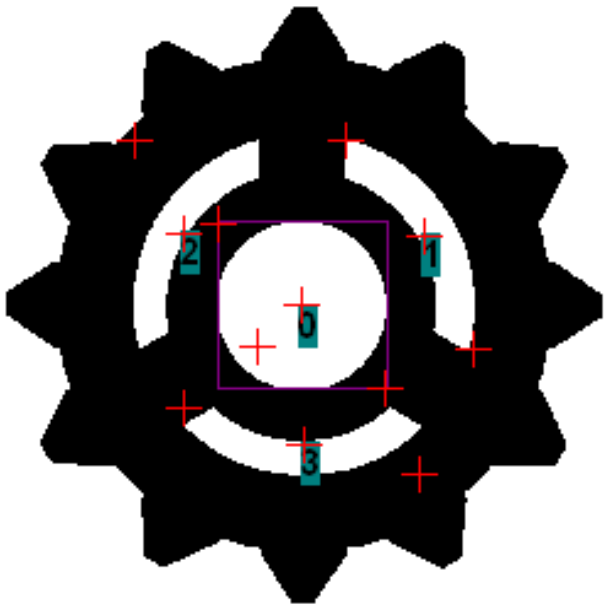
- Criar a subrotina GetInternalDiameter
- Criar uma ROI spoke devidamente centrada e dimensionada:
  - Impor as coordenadas com IO:ROI -> set coord
  - A ROI do tipo "Spoke" precisa de 3 pontos para se definir:
    - Centro, limite menor, limite maior
    - NB. Para impor a ROI é preciso ter os 3 pontos num array!
- Aplicar algoritmo de "Find Edge"
- Obter o melhor círculo a partir dos pontos de "Edge"

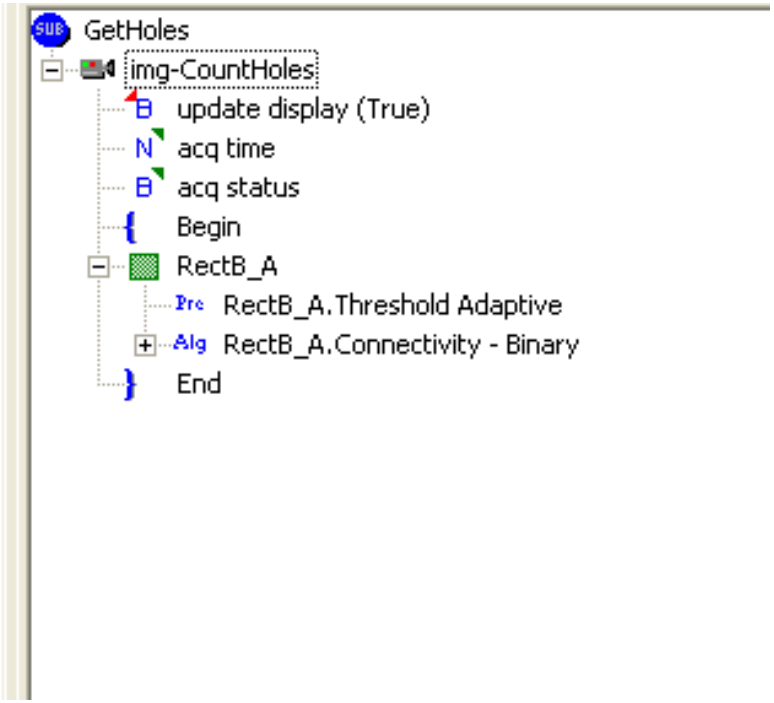




Exercício 6) – Número de Furos

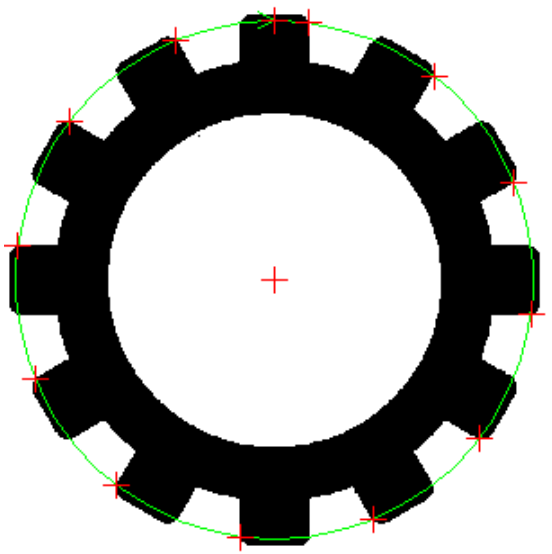
- Criar a subrotina GetHoles
- Numa cópia da imagem, usar o connectivity-binary para calcular o número de furos.

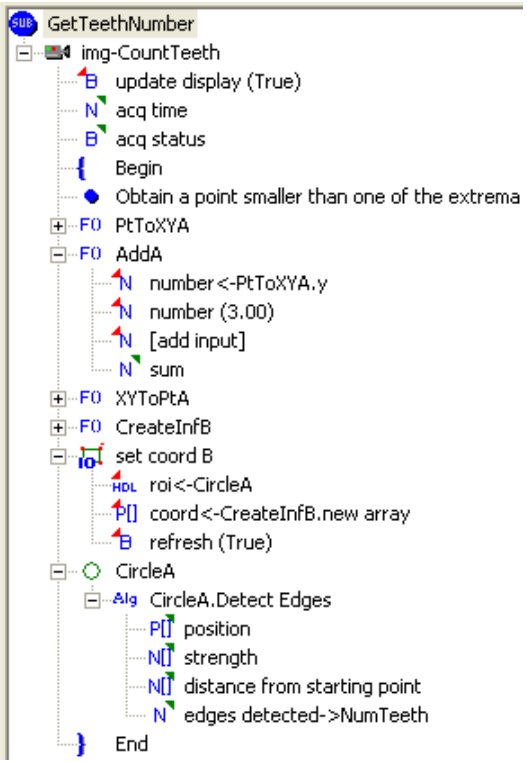




Exercício 7) – Contar número de dentes

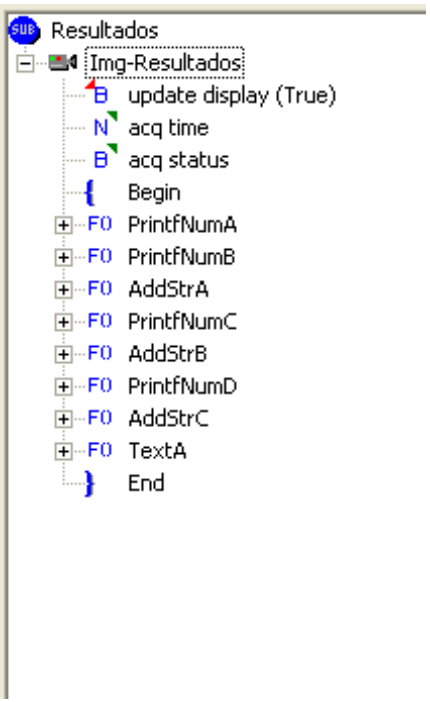
- Criar a subrotina GetTeethNumber
- Criar uma ROI circular devidamente centrada de forma a intercetar os dentes uma vez cada um.
- IO: ROI set coord – são necessários dois pontos para definir a ROI:
  - Centro, limite exterior (devem estar num array)
  - Sugestão: fazer o limite exterior 3 pixels menor que o diâmetro!
- Aplicar o algoritmo de "Detect Edges".



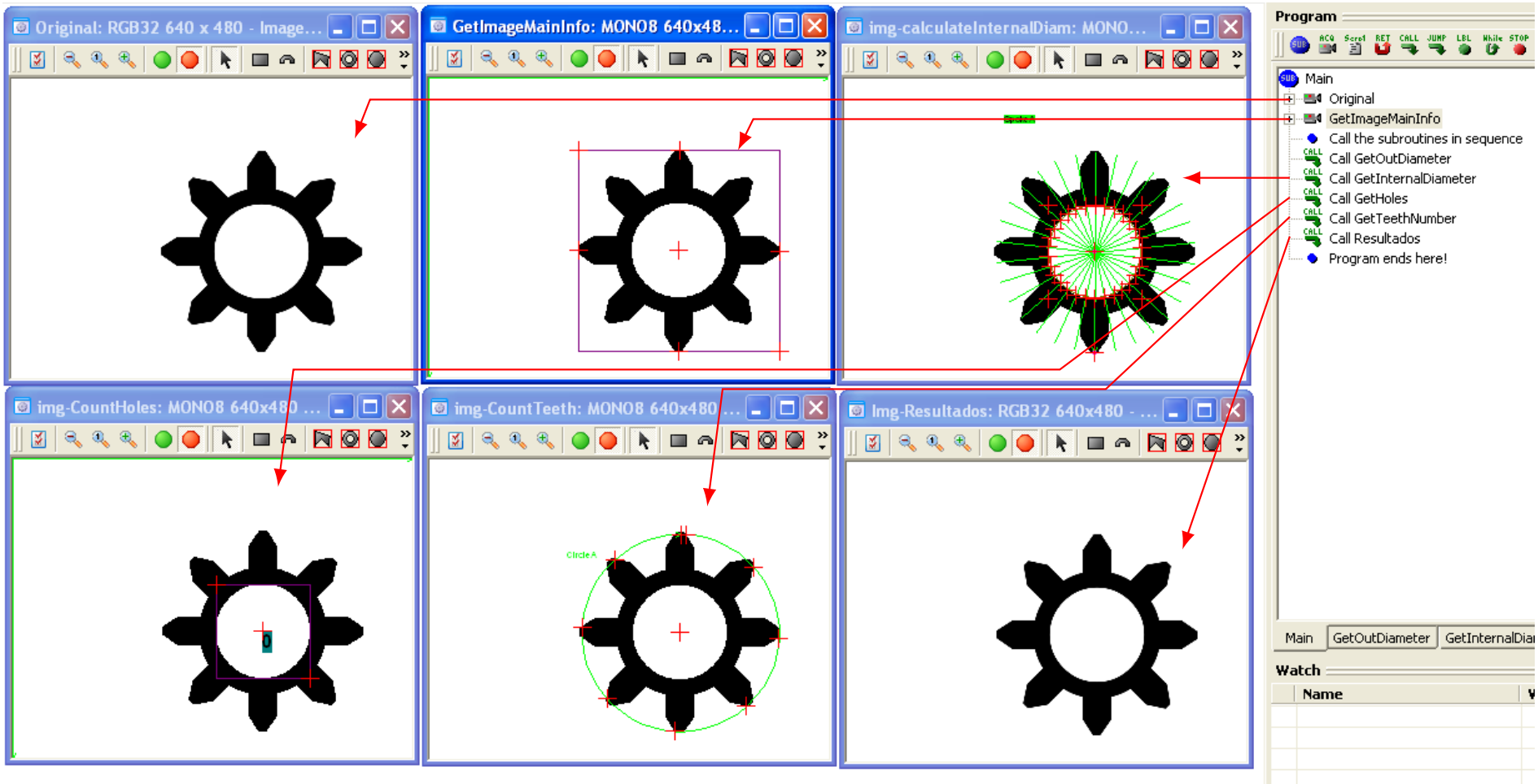


Exercício 8) – Escrever resultados na janela

- Criar a subrotina Resultados
- Gerar as strings adequadas e concatená-las de forma apropriada
- Apresentar o texto na imagem (com Text em IO:Annotation)



Um exemplo de layout da aplicação



- NB: Há muitas formas de resolver o problema. Esta é apenas uma das possibilidades de ferramentas a usar, e da organização da solução.