Universidade Federal de Campina Grande
IEEE Student Branch

IEEE Robotics & Automation Society

Capacitação Ras OnBoarding - Missão 02

O projeto RAS OnBoarding é a capacitação anual dos voluntários RAS UFCG e tem por objetivo instruir os voluntários sobre as principais áreas da robótica - eletrônica, visão computacional e simulação. Ele acontecerá em três etapas mensais, onde os participantes irão desenvolver missões com atividades semanais e irão apresentar aos respectivos coordenadores de suas equipes. Ao final do do projeto, espera-se que os voluntários tenham conhecimentos para desenvolverem seus próprios robôs e desempenharem com sucesso em novos projetos mais avançados.

Este documento trata sobre a **terceira atividade da Missão 02.** A Missão 02 se refere ao estudo dos principais algoritmos de visão computacional e suas aplicações na robótica.

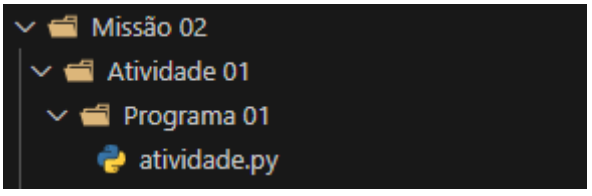
Palavras-chave: Visão Computacional, Python, OpenCV, Imagem, Vídeo, Espaço de cor, RGB, HSV, Detecção, Blobs, Contornos, Tags, Câmeras, Modelos de câmera, Segmentação, Filtros.

3º Atividade: Processamento de Imagens e Rastreio de objetos

Descrição: Utilizar a biblioteca do OpenCV para resolver problemas de visão computacional. Escolha **3 problemas** listados abaixo para solucionar:

Tarefas:

1. **Preparar um novo diretório para a atividade**. Durante esta atividade, serão desenvolvidos diversos programas e implementações de algoritmos em código. Crie a respectiva pasta desta atividade e faça o upload no repositório Github criado na atividade passada



Problemas a serem solucionados

1. **Reconhecimento de Gestos com a mão**
    a. Nesta tarefa você irá desenvolver um sistema que identifique gestos específicos feitos com as mãos, como sinais de "ok", "joinha", ou "pare".
    b. Objetivo: Aplicar detecção de corners para identificar as articulações dos dedos, detecção de blobs para segmentar a mão, e estimação de centróides para determinar a posição da mão na imagem.

2. **Calibração de Câmeras**
    a. Nesta tarefa você desenvolverá um programa capaz de calibrar uma câmera.
    b. Utilize o programa disponibilizado em Pattern Generation e crie o seu próprio padrão de calibração.
    c. Imprima o padrão de calibração que você gerou anteriormente e gere imagens do seu padrão em diferentes orientações.
    d. Utilize as imagens geradas para calibrar a câmera que você utilizou.
    e. Por fim, pesquise sobre as especificações da câmera que você utilizou e as compare com os dados obtidos na calibração. Trace conclusões.

3. **Estimação da Pose de um Robô usando Marcadores Fiduciais**
    a. Nesta tarefa você deve criar um sistema que utilize marcadores fiduciais para localizar e determinar a orientação de um robô em um ambiente controlado.
    b. Usar calibração de câmeras para compensar distorções, detecção de marcadores para identificar a posição do robô, e técnicas de estimação de centróides para calcular a orientação.
    c. O robô encontra-se disponível no laboratório. Imprima os marcadores fiduciais e use-os no robô

4. **Detecção de Padrões em Imagens de Rachaduras**
    a. Desenvolver um algoritmo que detecta padrões específicos em imagens de texturas, como detecção de rachaduras em pavimentos ou defeitos em superfícies metálicas.
    b. Utilizar detecção de corners para identificar bordas significativas e detecção de blobs para isolar áreas de interesse.
    c. Sugestão: utilize o dataset de imagens disponibilizado no Kaggle: Crack Segmentation Dataset.

5. **Reconhecimento de Formas em Imagens de Trânsito**
    a. Criar um sistema que identifique sinais de trânsito em imagens capturadas por uma câmera embarcada em um carro, reconhecendo placas de trânsito e as faixas da rodovia
    b. Aplicar detecção de corners e de linhas para definir as formas e detecção de blobs para isolar os sinais na imagem.
    c. Sugestão: utilize o dataset de imagens disponibilizado no Kaggle: Lane Detection Dataset.

Orientações: Deve-se trabalhar em grupo, lembrando que todos devem ter conhecimento das atividades desenvolvidas.

Responsáveis: Equipes/Todos.

Entrega: Apresentar o que foi desenvolvido em uma reunião semanal. Apenas uma por equipe.

Todas as atividades desenvolvidas devem ser apresentadas na reunião seguinte ao prazo final. O modelo de apresentação de slides do que foi desenvolvido durante o tempo proposto para a execução da atividade deve seguir o seguinte template:

- Template de Apresentação em LateX
- Guia Básico Prático - LateX

Obs.1: Todos que forem utilizar o link editável compartilhado deve realizar uma cópia pessoal para que não altere o modelo disponibilizado.

Obs.2: Todos que desejarem utilizar alguma outra ferramenta para a construção dos slides devem seguir o mesmo template, isto é, adaptar o template para a ferramenta desejada.

Data de Entrega: 27/08/2024

Material de Apoio: Não se limitem apenas aos links abaixo!

1. **Visão Computacional**
    - Notas de Aula USP - Visão Computacional
    - O que é VISÃO COMPUTACIONAL? Conceito e Aplicações | IA Descomplicada #02

2. **OpenCV**
    - OpenCV-Python Tutorials
    - OpenCV Course - Full Tutorial with Python
    - Como Controlar WebCam com Python [Introdução ao OpenCV]
    - Simple Color recognition with Opencv and Python
    - OpenCV Python Tutorial #5 - Colors and Color Detection

- #27 OPENCV - PYTHON | Image Thresholding | Global + Adaptive | ADAPTIVE MEAN & GAUSSIAN | Variants
- Tracking com Ajuste Dinâmico de Cores - Python & OpenCV 4 #08
- Intro and loading Images - OpenCV with Python for Image and Video Analysis 1
- Blurring and Smoothing - OpenCV with Python for Image and Video Analysis 8
- opencv project | opencv object tracking | lucas kanade | optical flow
- Blob Detection
- Blob Detection in OpenCV & Python: A Comprehensive Guide
- 11.7: Computer Vision: Blob Detection - Processing Tutorial
- Optical Flow for Object Tracking and Trajectories in OpenCV Python
- Pose Estimation of Objects in OpenCV Python
- Camera Calibration in Python with OpenCV - Python Script with Images
- DIY: OpenCV C# Find Specific Object & Pose Tracking
- Aruco Markers Tutorial
- 3. Object Tracking and Attitude / Pose Estimation Using Homography + PnP (OpenCV + Python)
- OpenCV orientation
- Learn Camera Calibration in OpenCV with Python Script
- Measure the size of an object | with Opencv, Aruco marker and Python

Mateus Pincho de Oliveira
Chair do Capítulo IEEE RAS UFCG
Campina Grande, 14 de agosto de 2024.