

# GARRA ROBÓTICA

## Uma garra robótica operada por gestos aprendidos utilizando aprendizagem profunda

### Fase 1 Implementação: Capturando e salvando gestos rotulados

Este projeto permite aos usuários controlar uma garra robótica utilizando gestos mostrados a uma câmera web.

#### 1 - Como funciona

O projeto é dividido em 3 fases, contidas em 3 cadernos jupyter, para atender os requisitos de usuário:

- Fase 1: Imagens devem ser capturadas com uma câmera web para compor um conjunto de gestos rotulados. O conjunto vai alimentar os conjuntos de treino e teste a serem utilizados em aprendizagem supervisionada.
- Fase 2: Um modelo de aprendizagem profunda, basicamente uma rede neural, será criado em utilizadopara treinar o reconhecimento de gestos, utilizando keras e tensorflow.
- Fase 3: Um programa será utilizado para ir capturando imagens de uma câmera web sequencialmente. As imagens serao classificadas utilizando o medelo treinado na Fase 2, e o resultado será utilizado para operar a garra robótica.

**Este caderno implementa a Fase 1 do pojeta. Existem outros dois cadernos a serem executados após este.**

In [ ]:

```
%load_ext autoreload
%autoreload 2
```






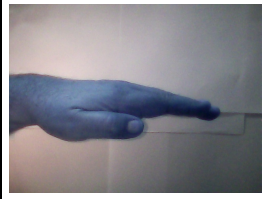



## 2 - Capturando imagens de gestos rotulados

As imagens serão capturadas da câmera web. Uma pasta com o nome **capture** vai possuir várias subpastas. As subpastas terão nomes significativos, como **left** (esquerda), **right** (direita), e assim por diante. A subpasta nomeada **left** vai conter imagens do gesto que lança o comando **turn to the left** (gire para a esquerda). A idéia é que os nomes das subpastas vão se tornar os valores das assertivas verdadeiras dos conjuntos para o processo de aprendizagem de máquina.

Para controlar a garra robótica, vamos utilizar nove comandos:

1. nothing (nada)
2. left (esquerda)
3. right (direita)
4. up (cima)
5. down (baixo)
6. foward (frente)
7. back (atrás)
8. grip (pegar)
9. loose (largar)

Alguns exemplos de imagens:

				
nothing (nada)	left (esqueda)	right (direita)	grip (pegar)	loose (largar)
				
foward (frente)	back (atrás)	up (cima)	down (baixo)	

## Packages (pacotes)

In [ ]:

```
%pylab inline
import cv2
from IPython.display import clear_output
import time
from datetime import datetime
import os
import numpy as np
```

A função **start\_webcam\_capture** recebe dois parametros, **path** e **number\_of\_captures** (10 por padrão).

- **path**: determina aonde em sua mídia as imagens capturadas serão armazenadas, i.e., `path = 'capture/left'`, onde a subpasta 'left' armazena os gestos rotulados 'left'.
- **number\_of\_captures**: é o número de imagens a serem capturadas e salvas.
- Pode-se interromper o kernel do caderno a fim de parar o código rodando na célula atual, mas frequentemente essa ação danifica a memória geral do caderno, e então é preciso usar o menu principal na sequência **menu> Kernel> Restart & Clear Output**, e então rodar todas as células (`ctrl + enter`) desde o início. As imagens capturadas antes da interrupção permanecem salvas, e como 'timestamps' são utilizados para garantir unicidade no nome dos arquivos, eles não serão sobrescritos. Na verdade, a função pode ser chamada quantas vezes desejar-se, até obter a quantidade de imagens desejadas. Pode-se também, utilizando um navegador de arquivos, inspecionar as imagens capturadas e manualmente excluir algumas por ventura indesejadas.
- **return**: (retorno) Ao final a função informa quantas imagens estão armazenadas no 'path'.

In [ ]:

```

"""
    function start_webcam_capture
    parameters:
    path - O caminho onde as imagens capturadas serão salvas (o nome da última pasta é usado no futuro para rotular as imagens)
    number_of_captures - o numero de imagens a serem capturadas a cada execução da função.
"""
def start_webcam_capture(path, number_of_captures=10):
    # variáveis para definir detalhes do som de aviso da captura (é necessário que o programa play esteja instalado no s.o.)
    frequency = 100 # Hertz
    duration = 50 # milliseconds
    #vamos garantir que o 'path' exista!
    if not os.access(path, os.F_OK):
        os.makedirs(path)
    count_captures = 0
    #utilizando a câmera web 0.
    #em alguns sistemas a câmera desejada pode estar em diferentes numeros, i.e, 1 ou 2 ou 3 ...
    ##### câmera a utilizar #####
    vid = cv2.VideoCapture(0)
    #####
    start_time = time.time()
    try:
        while(count_captures<number_of_captures):
            # Captura quadro-a-quadro
            ret, frame = vid.read()
            if not ret:

```

```

        # Liberar o dispositivo de vídeo (Video Device) se ret for falso
        vid.release()
        # Mensagem a ser mostrada após liberar o dispositivo de vídeo
        print("Recurso de vídeo liberado devido a falha na captura, veri-
fique sua câmera e tente de novo!")
        break
        # Converter a imagem de formato OpenCV BGR para formato matplotlib R
GB
        # para poder mostrar a imagem na tela
        frame = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
        # verificar se é hora de salvar um quadro em um arquivo (ex.: a cada
4 segundos)
        elapsed_time = time.time() - start_time
        if elapsed_time > 4:
            # tocar som para indicar a ação
            os.system('play -n synth %s sin %s' % (duration/1000, frequency
))
            timestamp = datetime.utcnow().strftime('%Y_%m_%d_%H_%M_%S_%f')[:
-3]

            timestamp = timestamp + '.jpg'
            image_filename = os.path.join(path, timestamp)
            #print(image_filename)
            cv2.imwrite(image_filename, frame)
            #incrementar count_captures
            count_captures += 1
            #reiniciar o timer
            start_time = time.time()
        # verificar se a tecla ESC foi acionada
        key = np.int16(cv2.waitKey(1))
        if key == 27:
            print("Esc key interrupted!")
            break # esc para parar, mas, ATT: NÃO FUNCIONA EM CADERNO JUPYT
ER :(
        # esconde os eixos (axis)
        axis('off')
        # Titulo da janela
        title("Captura de Gestos para a Garra Robótica")
        # Mostra o quadro
        imshow(frame)
        show()
        # Permanece mostrando até novo quadro estar disponível
        clear_output(wait=True)
    except KeyboardInterrupt:
        # Mensagem a ser exibida após liberar o dispositivo
        print("keyboard interrupted!")
        # Liberar o dispositivo
        vid.release()
        print("Liberado o dispositivo de video")
        path, dirs, files = os.walk(path).__next__()
        file_count = len(files)
        print('Existem agora ', file_count, ' imagens em ', path)

```

Vamos iniciar a captura de gestos para **nothing** (nada). Vamos executar a função com os valores de parâmetros desejados.

In [ ]:

```
path = 'capture/nothing'  
#inicia a captura das imagens de gestos  
start_webcam_capture(path)
```

Vamos capturar gestos para **left** (esquerda).

In [ ]:

```
path = 'capture/left'  
#inicia a captura das imagens de gestos  
start_webcam_capture(path)
```

Vamos capturar gestos para **right** (direita).

In [ ]:

```
path = 'capture/right'  
#inicia a captura das imagens de gestos  
start_webcam_capture(path)
```

Vamos capturar gestos para **up** (cima).

In [ ]:

```
path = 'capture/up'  
#inicia a captura das imagens de gestos  
start_webcam_capture(path)
```

Vamos capturar gestos para **down** (baixo).

In [ ]:

```
path = 'capture/down'  
#inicia a captura das imagens de gestos  
start_webcam_capture(path)
```

Vamos capturar gestos para **foward** (frente).

In [ ]:

```
path = 'capture/foward'  
#inicia a captura das imagens de gestos  
start_webcam_capture(path)
```

Vamos capturar gestos para **back** (atrás).

In [ ]:

```
path = 'capture/back'  
#inicia a captura das imagens de gestos  
start_webcam_capture(path)
```

Vamos capturar gestos para **grip** (segurar).

In [ ]:

```
path = 'capture/grip'  
#inicia a captura das imagens de gestos  
start_webcam_capture(path)
```

Vamos capturar gestos para **loose** (largar).

In [ ]:

```
path = 'capture/loose'  
#inicia a captura das imagens de gestos  
start_webcam_capture(path)
```

Aquí termina este caderno. Agora deve existir em seu caminho de captura 9 subpastas (nothing, left, right, up, down, foward, back, grip and loose), cada uma com várias imagens. Em seguida, desejamos construir e treinar um modelo para reconhecer gestos, utilizando os dados obtidos. Esta vai ser a tarefa do próximo caderno, **02\_treinado\_o\_modelo\_garraRobotica**.

by Duodecimo, 2017, Dezembro.