Let's play with Python and OpenCV

Omar Trinidad Gutiérrez Méndez

División Académica de Informática y Sistemas
Universidad Juárez Autónoma de Tabasco

July 6, 2012

Computer Vision

Computer Vision

A picture is worth a thousand words.

Computer Vision

A picture is worth a thousand words.

- The computer vision is the science and engineering discipline concerned with making inferences about the external world.

Computer Vision

A picture is worth a thousand words.

- The computer vision is the science and engineering discipline concerned with making inferences about the external world.
- What is an image? Is an array. An array of pixels.

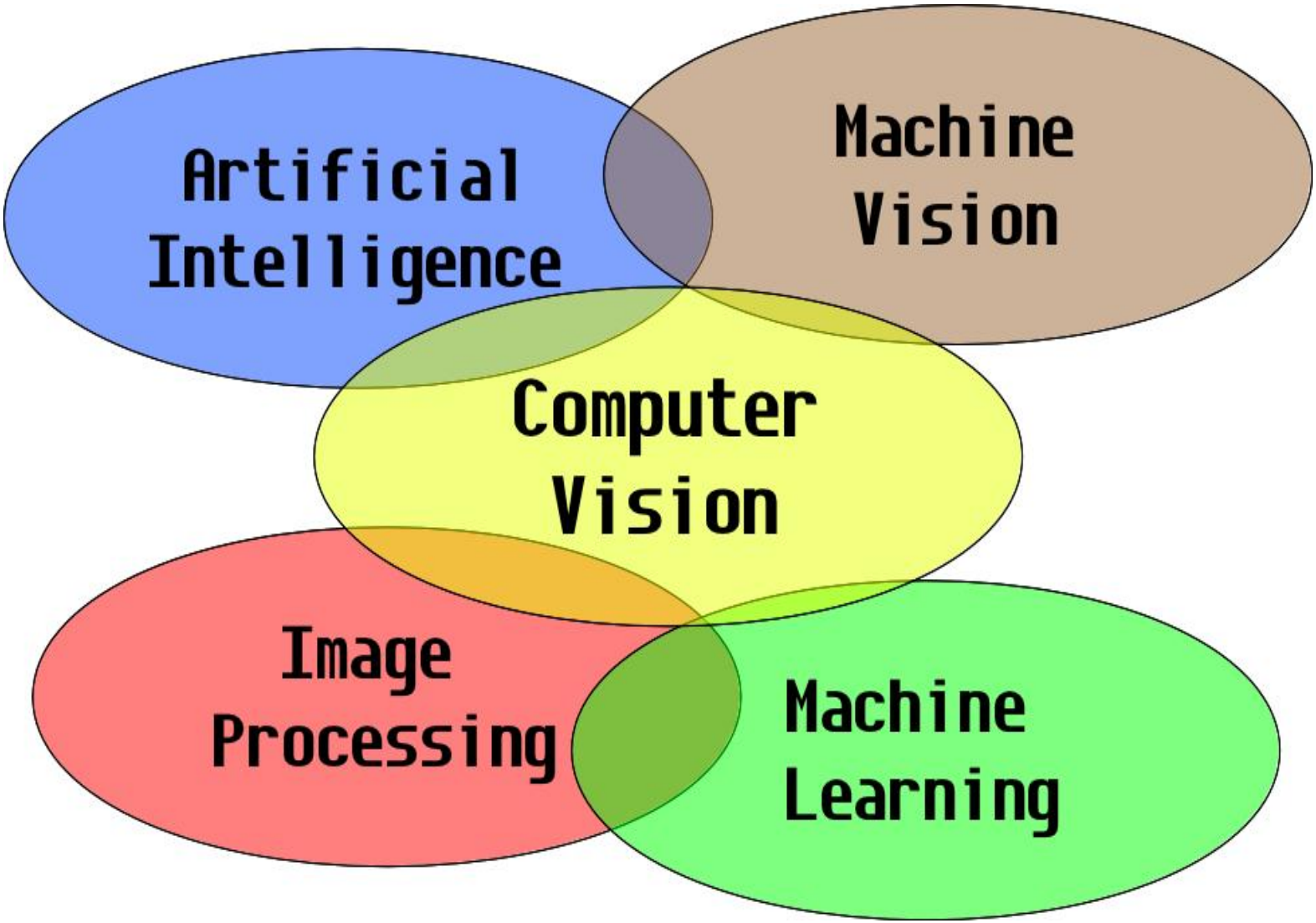
Computer Vision

A picture is worth a thousand words.

- The computer vision is the science and engineering discipline concerned with making inferences about the external world.
- What is an image? Is an array. An array of pixels.
- The goal of the computer vision is to achieve something similar to the human perception.

Computer Vision

Computer Vision



OpenCV

OpenCV

- OpenCV (Open Source Computer Vision) is a library of programming functions for real time computer vision.

OpenCV

- OpenCV (Open Source Computer Vision) is a library of programming functions for real time computer vision.
- Is realeased under a BSD license.

OpenCV

- OpenCV (Open Source Computer Vision) is a library of programming functions for real time computer vision.
- Is realeased under a BSD license.
- Languages:

OpenCV

- OpenCV (Open Source Computer Vision) is a library of programming functions for real time computer vision.
- Is realeased under a BSD license.
- Languages:
  - C

OpenCV

- OpenCV (Open Source Computer Vision) is a library of programming functions for real time computer vision.
- Is realeased under a BSD license.
- Languages:
  - C
  - C++

OpenCV

- OpenCV (Open Source Computer Vision) is a library of programming functions for real time computer vision.
- Is realeased under a BSD license.
- Languages:
  - C
  - C++
  - Java

OpenCV

- OpenCV (Open Source Computer Vision) is a library of programming functions for real time computer vision.
- Is realeased under a BSD license.
- Languages:
  - C
  - C++
  - Java
  - Python ;)

OpenCV

- OpenCV (Open Source Computer Vision) is a library of programming functions for real time computer vision.
- Is realeased under a BSD license.
- Languages:
  - C
  - C++
  - Java
  - Python ;)
- Operating systems:

OpenCV

- OpenCV (Open Source Computer Vision) is a library of programming functions for real time computer vision.
- Is realeased under a BSD license.
- Languages:
  - C
  - C++
  - Java
  - Python ;)
- Operating systems:
  - Windows

OpenCV

- OpenCV (Open Source Computer Vision) is a library of programming functions for real time computer vision.
- Is realeased under a BSD license.
- Languages:
  - C
  - C++
  - Java
  - Python ;)
- Operating systems:
  - Windows
  - Mac

OpenCV

- OpenCV (Open Source Computer Vision) is a library of programming functions for real time computer vision.
- Is realeased under a BSD license.
- Languages:
  - C
  - C++
  - Java
  - Python ;)
- Operating systems:
  - Windows
  - Mac
  - Linux

OpenCV

- OpenCV (Open Source Computer Vision) is a library of programming functions for real time computer vision.
- Is realeased under a BSD license.
- Languages:
  - C
  - C++
  - Java
  - Python ;)
- Operating systems:
  - Windows
  - Mac
  - Linux
  - Android and iOS

OpenCV main versions

OpenCV main versions

- Version 1.0

OpenCV main versions

- Version 1.0
- Version 2.X.X

OpenCV main versions

- Version 1.0
- Version 2.X.X
- Version 3.X.X

Binding cv versus cv2

Binding cv versus cv2

- Binding cv

Binding cv versus cv2

- Binding cv
  - Is based in ANSI C

Binding cv versus cv2

- Binding cv
  - Is based in ANSI C
  - Is tricky

Binding cv versus cv2

- Binding cv
  - Is based in ANSI C
  - Is tricky
  - Uses IPLImage objects like images

Binding cv versus cv2

- Binding cv
  - Is based in ANSI C
  - Is tricky
  - Uses IPLImage objects like images
- Binding cv2

Binding cv versus cv2

- Binding cv
  - Is based in ANSI C
  - Is tricky
  - Uses IPLImage objects like images
- Binding cv2
  - Is based in C++

Binding cv versus cv2

- Binding cv
  - Is based in ANSI C
  - Is tricky
  - Uses IPLImage objects like images
- Binding cv2
  - Is based in C++
  - Is more friendly ;)

Binding cv versus cv2

- Binding cv
  - Is based in ANSI C
  - Is tricky
  - Uses IPLImage objects like images
- Binding cv2
  - Is based in C++
  - Is more friendly ;)
  - Is fastest

Binding cv versus cv2

- Binding cv
  - Is based in ANSI C
  - Is tricky
  - Uses IPLImage objects like images
- Binding cv2
  - Is based in C++
  - Is more friendly ;)
  - Is fastest
  - Uses NumPy like images

Basic functions

Basic functions

- `cv2.imread(path) -> retval`

Basic functions

- `cv2.imread(path) -> retval`
- `cv2.namedWindow(name) -> None`

Basic functions

- `cv2.imread(path) -> retval`
- `cv2.namedWindow(name) -> None`
- `cv2.destroyWindow(name) -> None`

Basic functions

- `cv2.imread(path) -> retval`
- `cv2.namedWindow(name) -> None`
- `cv2.destroyWindow(name) -> None`
- `cv2.imshow(title, image)`

Basic functions

- `cv2.imread(path) -> retval`
- `cv2.namedWindow(name) -> None`
- `cv2.destroyWindow(name) -> None`
- `cv2.imshow(title, image)`
- `cv2.imwrite(path, image)`

Basic functions

- `cv2.imread(path) -> retval`
- `cv2.namedWindow(name) -> None`
- `cv2.destroyWindow(name) -> None`
- `cv2.imshow(title, image)`
- `cv2.imwrite(path, image)`
- `cv2.waitKey(time)`

Basic functions

- `cv2.imread(path) -> retval`
- `cv2.namedWindow(name) -> None`
- `cv2.destroyWindow(name) -> None`
- `cv2.imshow(title, image)`
- `cv2.imwrite(path, image)`
- `cv2.waitKey(time)`
- `cv2.startWindowThread()`

Basic filters

Some basic filters

Some basic filters

- `cv2.blur(image, kernel) -> image`

Some basic filters

- `cv2.blur(image, kernel) -> image`
- `cv2.Laplacian(image, depth) -> image`

Some basic filters

- `cv2.blur(image, kernel) -> image`
- `cv2.Laplacian(image, depth) -> image`
- `cv2.cvtColor(image, code*) -> image`

Some basic filters

- `cv2.blur(image, kernel) -> image`
- `cv2.Laplacian(image, depth) -> image`
- `cv2.cvtColor(image, code*) -> image`
- `cv2.threshold(image, threshold, maxval, type*) -> image`

Some basic filters

- `cv2.blur(image, kernel) -> image`
- `cv2.Laplacian(image, depth) -> image`
- `cv2.cvtColor(image, code*) -> image`
- `cv2.threshold(image, threshold, maxval, type*) -> image`
- `cv2.dilate(image, kernel) -> image`

Some basic filters

- `cv2.blur(image, kernel) -> image`
- `cv2.Laplacian(image, depth) -> image`
- `cv2.cvtColor(image, code*) -> image`
- `cv2.threshold(image, threshold, maxval, type*) -> image`
- `cv2.dilate(image, kernel) -> image`
- `cv2.erode(image, kernel) -> image`

Some basic filters

- `cv2.blur(image, kernel) -> image`
- `cv2.Laplacian(image, depth) -> image`
- `cv2.cvtColor(image, code*) -> image`
- `cv2.threshold(image, threshold, maxval, type*) -> image`
- `cv2.dilate(image, kernel) -> image`
- `cv2.erode(image, kernel) -> image`
- `cv2.getStructuringElement(shape*, size) -> structure`

A very simple camera

A very simple camera

```
1 import cv2
2 cam = cv2.VideoCapture(0)
3 while True:
4   img = cam.read()[1]
5   cv2.imshow("Window", img)
6   if cv2.waitKey(5) == 32:
7     break
```

An example with FITS files

An example with FITS files

- We will play with stars ;)
  - The first step is load a FITS file.
  - Apply `cvtColor` and `threshold` functions to the image.
  - Use `floodFill` function to coloring the stars.
  - Draw rectangles in the objects.
  - Slow?

An example with FITS files

- We will play with stars ;)
  - The first step is load a FITS file.
  - Apply `cvtColor` and `threshold` functions to the image.
  - Use `floodFill` function to coloring the stars.
  - Draw rectangles in the objects.
  - Slow? Ha', very slow.

Good idea and bad idea

Good idea and bad idea

- Bad idea: Traverse all the items in an array.

```
1 for x in xrange(height):
2   for y in xrange(width):
3     image.itemset(x, y, 0, data.item(x,y))
```

- Worst idea: Use indexing syntax.

```
1 for x in xrange(height):
2   for y in xrange(width):
3     image[x, y] = data[x,y]
```

- Good ideas:
  - If you feel the need for speed, go for built-in functions. *Guido Van Rossum*.
  - Read `An Optimization Anecdote` in `http://www.python.org/doc/essays/list2str/`

Searching the yellow color

Searching the yellow color

- Use the simple camera and apply the blur filter.
- Convert the image to HSV color model and get a kind of threshold (color detection).
- Get the moments of the image
- Draw anything

OpenCV today

- OpenCV 2.4.2 has been released (two days ago)

OpenCV today

- OpenCV 2.4.2 has been released (two days ago)
- Support for CUDA NVIDIA

OpenCV today

- OpenCV 2.4.2 has been released (two days ago)
- Support for CUDA NVIDIA
- Support for Android

OpenCV today

- OpenCV 2.4.2 has been released (two days ago)
- Support for CUDA NVIDIA
- Support for Android
- Support for iOS

OpenCV today

- OpenCV 2.4.2 has been released (two days ago)
- Support for CUDA NVIDIA
- Support for Android
- Support for iOS
- Each version of OpenCV Python wrapper is more *pythonic*

Thanks!

About me

@omar_trinidad

omar.vpa@gmail.com

Questions?