Visão Computacional: Aplicações de Inteligência Artificial com poucas linhas de código

Wolfram Cloud & Wolfram Language

Semana da Engenharia - UNICID - 2015 - SP - Brasil

- Daniel Carvalho @danielscarvalho - Advisor Tecnologia
- Adriano Tegani @ategani - Advisor Tecnologia
- Marcelo Vianello - UNICID

http://www.wolframcloud.com/
http://www.advisor.net.br/
http://www.unicid.com.br/

Conceitos

Visão Computacional e Inteligência Artificial com o software *Mathematica*

Como obter informação a partir de imagens com poucas linhas de código

- Programação funcional
- Linguagem de quarta geração (mais alto nível)
- Inteligência Artificial
- Morfologia Matemática (Morphology)
- Computação Simbólica

A Inteligência artificial (IA) está presente em nosso dia a dia, os serviços mais utilizados e interessantes na Internet atualmente fazem uso da IA tal como: Google, Youtube, Amazon, Netflix, Facebook, Twitter entre outros.

Com o avanço da IA há ameaça dos trabalhos em escritório (colarinho branco - white-collar). Assim como os robôs substituíram os operários na indústria (colarinho azul - blue-collar), a AI vai automatizar muitas das atividades de escritório em curto e médio prazo. Porém novas profissões e demandas de mercado serão geradas, novas profissões que não temos consciência ainda vão surgir.

A sorte favorece a mente bem preparada.
Louis Pasteur

Mathematica & Wolfram Language

Mais sobre Visualização de Dados e Visão Computacional:

Channels: Wolfram Language Image Processing Virtual Workshop
The Wolfram Language: FAST INTRODUCTION FOR PROGRAMMERS

Imagens provenientes de busca no “Google Images”, são de propriedade de seus respectivos autores!!! Uso didático, imagens utilizadas conforme preferência/conveniência da audiência da apresentação.

Os comandos do *Mathematica* são normalmente palavras em inglês com a primeira letra em caixa alta, os parâmetros são separados por vígula e ficam entre colchetes **[ ]**, listas (vetores e matrizes) são agrupados entre chaves **{0, 1, 2, 3}** e os itens separados por vírgula também. Parâmetros das funções seguem a forma “**NomeDoParâmetro -> Valor**”. Depois de digitar os comandos utilize as combinação de teclas **shift + enter** para processar/executar as instruções.

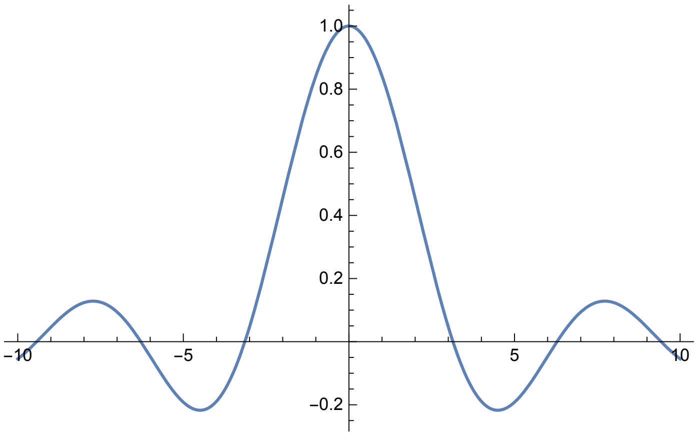
É fácil para iniciantes encontrar a função que lhes interessa pois estão relacionadas com seu objetivo, e são definidos no sistema de forma consistente.

```
2 + 2

4

Plus[2, 2]

4

1 / 9

1
—
9

N[1 / 9, 100]

0.1111111111111111111111111111111111111111111111111111111111111111111111111111⋱
 111111111111111111111111

MatrixForm[{{a, b, c}, {1, 2, 3}, {10, 20, 30}}]

( a   b   c  )
( 1   2   3  )
( 10  20  30 )

m1 = {{a, b, c}, {1, 2, 3}, {10, 20, 30}};

m2 = {i, j, k};

m2 * m1 // MatrixForm

( a i   b i   c i  )
(  j    2 j   3 j  )
( 10 k  20 k  30 k )
```

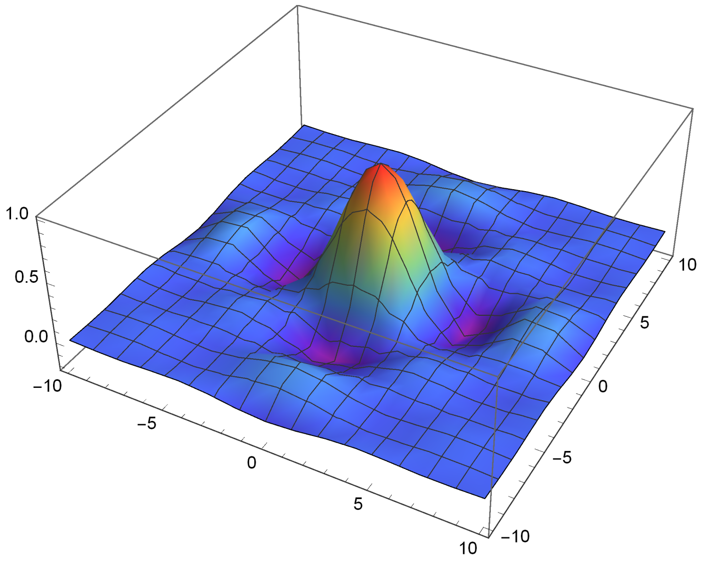
```
m2 . m1 // MatrixForm
```

$$\begin{pmatrix} a\,i + j + 10\,k \\ b\,i + 2\,j + 20\,k \\ c\,i + 3\,j + 30\,k \end{pmatrix}$$

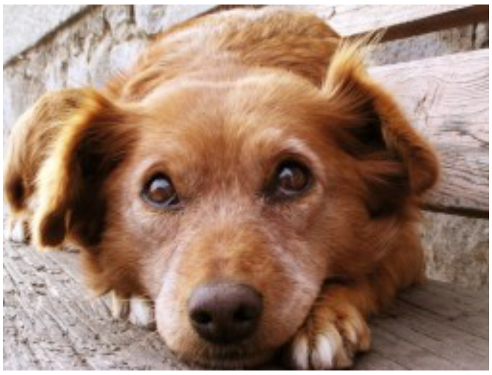
```
Plot[Sinc[x], {x, -10, 10}]
```



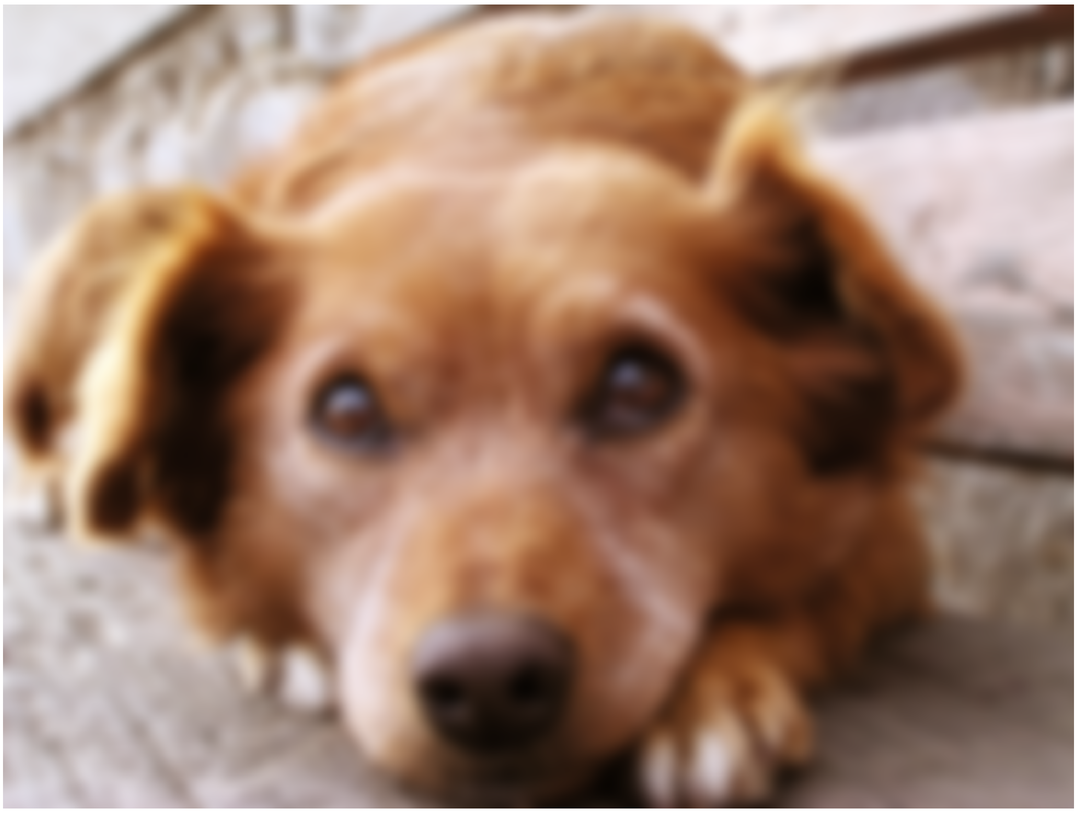
```
Plot3D[Sinc[x] * Sinc[y],
 {x, -10, 10},
 {y, -10, 10},
 PlotRange → All,
 ColorFunction → "Rainbow"]
```



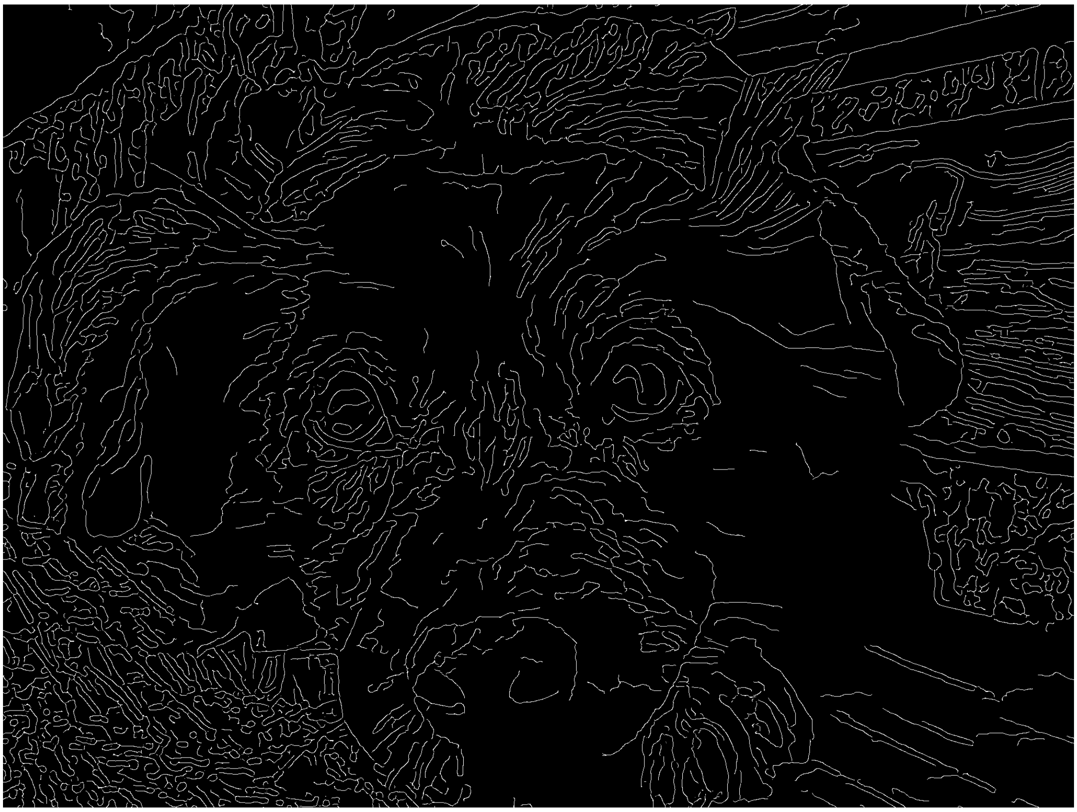
```
img1 = Import[
  "https://www.petfinder.com/wp-content/uploads/2012/11/dog-how-to-select-your-
    new-best-friend-thinkstock99062463-253x190.jpg"]
```



```
Blur[img1, 30]
```



```
EdgeDetect[img1, 10]
```



Pesquisa livre: em Inglês (free form input)

= sertaozinho to sao paulo

= show sky now

= what is the meaning of life?

= who you are?

= who is Sonia Braga?

Dados são obtidos do site: www.wolframalpha.com | Desenvolvido com Wolfram Language e *Mathematica*

barretos to sao paulo

↳ Distance

386.5 km (kilometers)

what is the meaning of life? »

↳ Result

42
(according to the book *The Hitchhiker's Guide to the Galaxy*, by Douglas Adams)

who are you? »

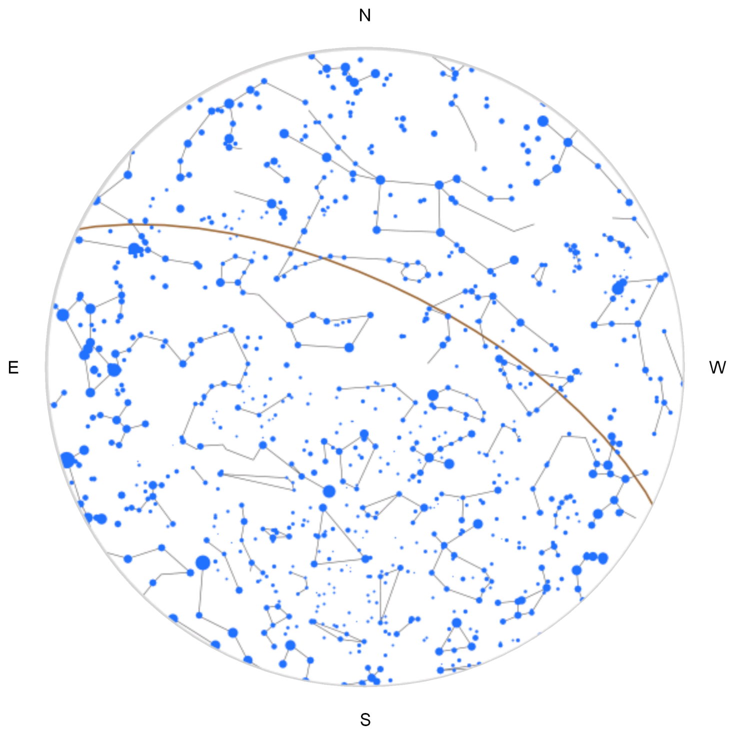
↳ Response

```
"My name is Wolfram|Alpha."
```

```
My name is Wolfram|Alpha.
```

show sky now »

Results (1 of 2)



who is Sonia Braga?

Sonia Braga (person)

Input interpretation:

Sonia Braga (person)

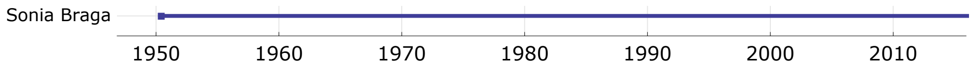
Sonia Braga (actor)

Basic information:

full name	Sônia Maria Campos Braga
date of birth	Thursday, June 8, 1950 (age: 65 years)
place of birth	Maringa, Parana

Image:

Timeline:



Familial relationships:

Parents:

Maria José Braga | Hélio Fernando Ferraz Braga

Siblings:

Ana Maria Braga | Hélio Braga | Maria Braga | Júlio Braga

Spouse:

Pat Metheny (domestic partnership)

Notable films: More

Appeared in:

Angel Eyes (2001) | Kiss of the Spider Woman (1985) | The Rookie (1990) | The Milagro Beanfield War (1988) | Empire (2002) | … (total: 14)

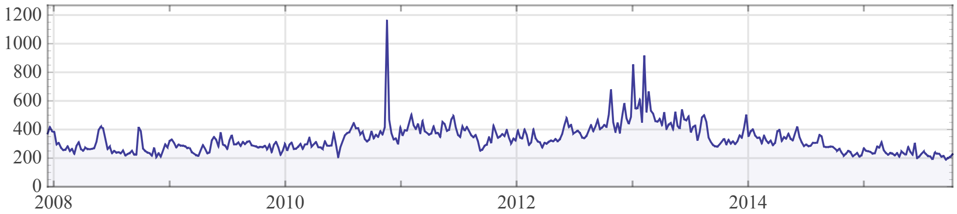
+ Definitions

Wikipedia summary:

Sônia Maria Campos Braga (born 8 June 1950) is a Brazilian actress. Nominated for three Golden Globes and an Emmy Award, Braga is best known for her performances in “ Kiss of the Spider Woman,” “ Dona Flor and Her Two Husbands ” and “ Moon Over Parador.” Her television credits include “ Sex and the City,” “ Alias,” “ American Family ” and "The Cosby Show".

Full entry »

Wikipedia page hits history: Log scale

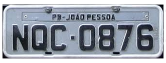


(in hits per day)

(based on weekly averages of daily hits to English-language "Sonia Braga" page)

Reconhecimento de placas de carro - OCR

Imagens de placas obtidas do Google Images: placas de carros

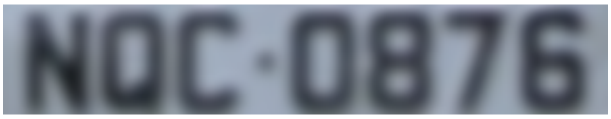
TextRecognize[ **]**

placas = { , , , ,

, , **};**

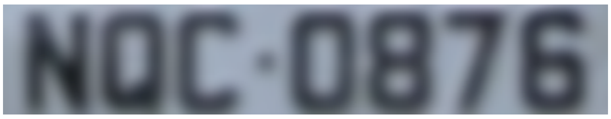
Blur[ImageTake[**, {40, 96}, {10, -10}], 8]**

TextRecognize[%]



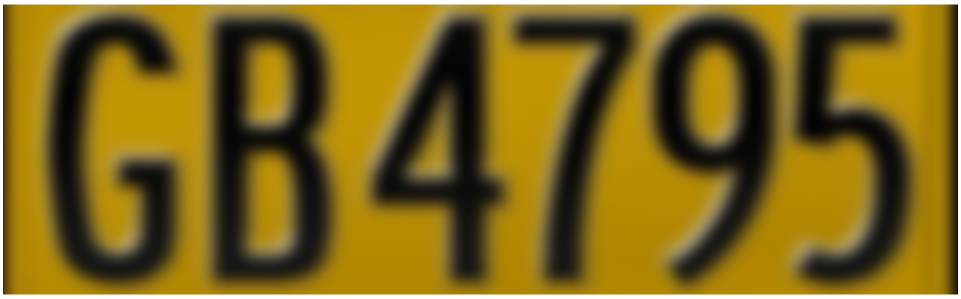
```
NOC 0875
```

Blur[ImageTake[**, {40, 96}, {10, -10}], 8]**



Blur[ImageTake[**, {70, 220}], 8]**

TextRecognize[%]



```
lGB4795|
```

TextRecognize[Blur[ImageTake[**, -160], 8]]**

```
[HQW-5678]
```

```
GetText[input_List] := TextRecognize[Blur[ImageTake[#, -160], 8]] & /@ input;
```

```
GetText[placas]
```

```
{[HQW~5(;78], ABZC- | 2i34|, ABC- |234, Ic»;1'5°50vf;|, G_B4/‘25|, , LHEEQ}
```

```
StringReplace[#, {"[" → "", "]" → "", " " → "", "|" → "1"}] & /@ GetText[placas]
```

```
{HQW-5678, ABZC-12i341, ABC-1234, Icnffiovrzl, G_B4/‘251, , LHEEQ}
```

Reconhecimento de face

```
family = ExampleData[{"TestImage", "Lena"}]
```

```
family = Import["lena.jpg"]
```

```
family = Import["the-simpsons.jpg"]
```

```
family = Import["monica.jpg"];
```

```
caras = FindFaces[family]
```

```
{{{215.5, 135.5}, {388.5, 308.5}}}
```

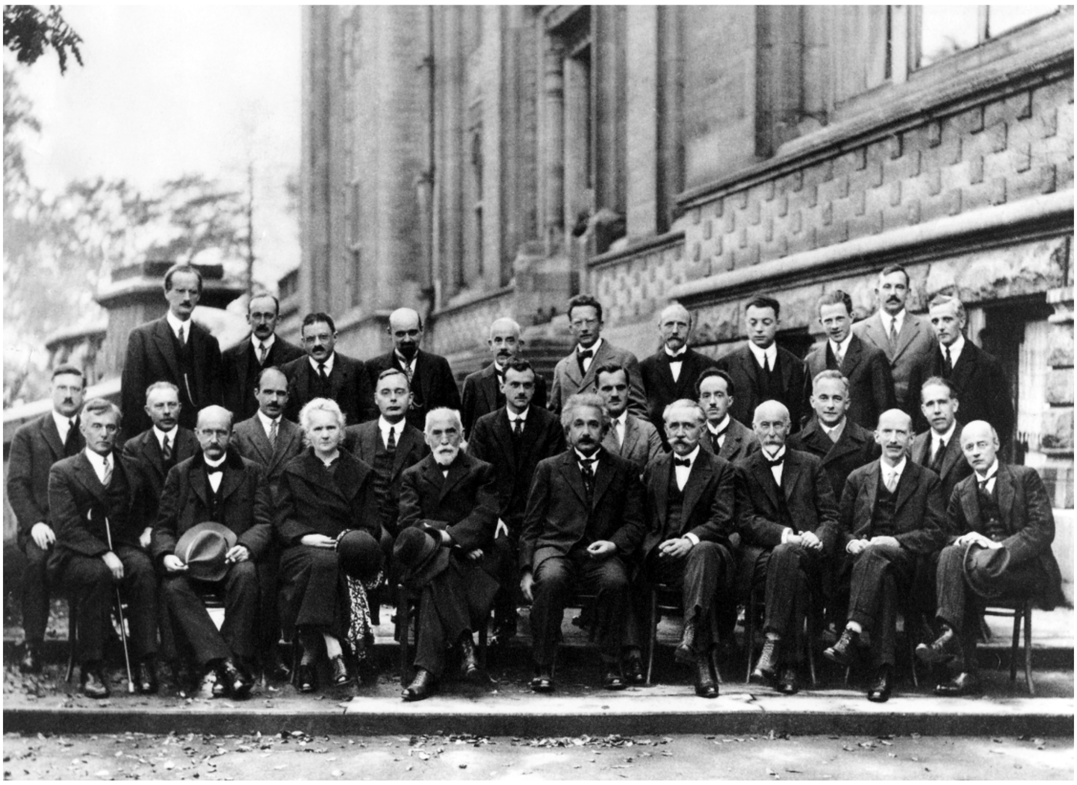
```
ImageTrim[family, #] & /@ caras
```

{ }

```
Show[family, Graphics[{EdgeForm[{Red, Thick}], Opacity[0], Rectangle @@@ caras}]]
```

IA não é robusta para reconhecer/distinguir desenhos!!! :-) Treinada apenas para rostos humanos. Podem ocorrer falsos positivos (overfitting).

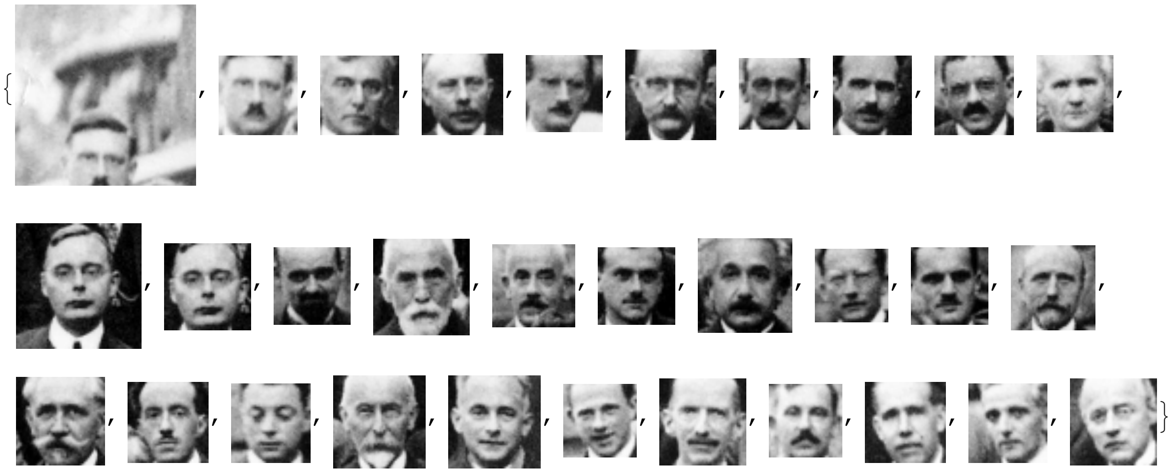
```
conference = Import["conference.jpg"]
```



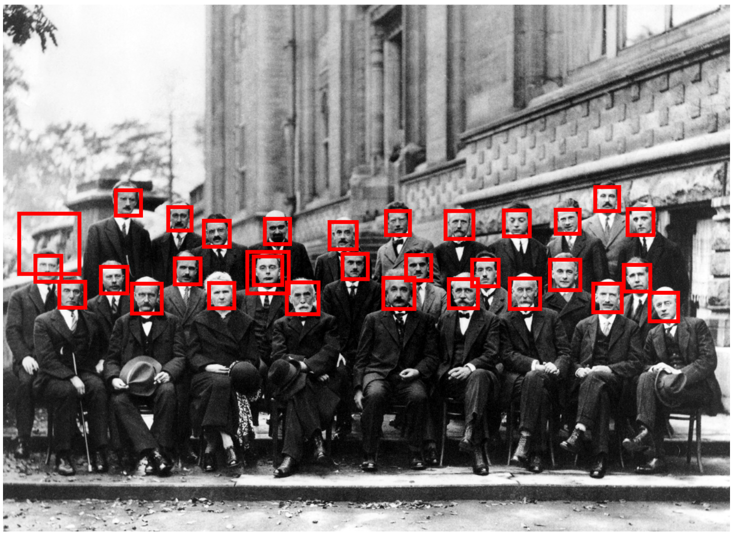
```
caras = FindFaces[conference]
```

```
{{{24.5, 396.5}, {117.5, 489.5}},
 {{49.5, 384.5}, {89.5, 424.5}}, {{86.5, 344.5}, {126.5, 384.5}},
 {{150.5, 366.5}, {191.5, 407.5}}, {{173.5, 486.5}, {212.5, 525.5}},
 {{198.5, 335.5}, {244.5, 381.5}}, {{254.5, 463.5}, {290.5, 499.5}},
 {{264.5, 380.5}, {304.5, 420.5}}, {{309.5, 438.5}, {349.5, 478.5}},
 {{317.5, 343.5}, {356.5, 382.5}}, {{376.5, 366.5}, {440.5, 430.5}},
 {{386.5, 379.5}, {430.5, 423.5}}, {{403.5, 442.5}, {442.5, 481.5}},
 {{437.5, 334.5}, {486.5, 383.5}}, {{503.5, 434.5}, {545.5, 476.5}},
 {{523.5, 388.5}, {562.5, 427.5}}, {{586.5, 342.5}, {634.5, 390.5}},
 {{590.5, 456.5}, {627.5, 493.5}}, {{621.5, 386.5}, {660.5, 425.5}},
 {{682.5, 450.5}, {725.5, 493.5}}, {{687.5, 343.5}, {732.5, 388.5}},
 {{723.5, 377.5}, {764.5, 418.5}}, {{772.5, 454.5}, {812.5, 494.5}},
 {{781.5, 342.5}, {828.5, 389.5}}, {{843.5, 371.5}, {890.5, 418.5}},
 {{852.5, 458.5}, {889.5, 495.5}}, {{910.5, 337.5}, {954.5, 381.5}},
 {{913.5, 493.5}, {950.5, 530.5}}, {{957.5, 369.5}, {998.5, 410.5}},
 {{963.5, 456.5}, {1003.5, 496.5}}, {{997.5, 323.5}, {1041.5, 367.5}}}
```

```
ImageTrim[conference, #] & /@ caras
```



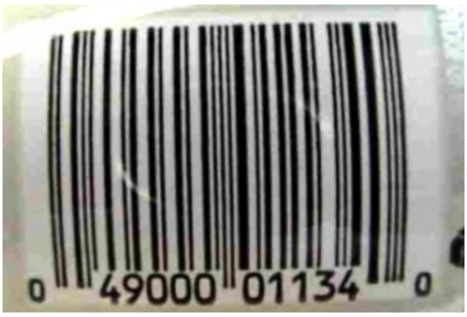
```
Show[conference,
 Graphics[{EdgeForm[{Red, Thick}], Opacity[0], Rectangle @@@ caras}]]
```



Há 1 falso positivo

Reconhecimento de código de barra

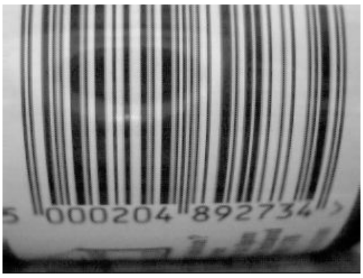
BarcodeRecognize[



]

```
049000011340
```

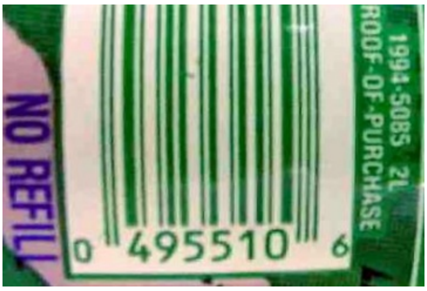
BarcodeRecognize[



]

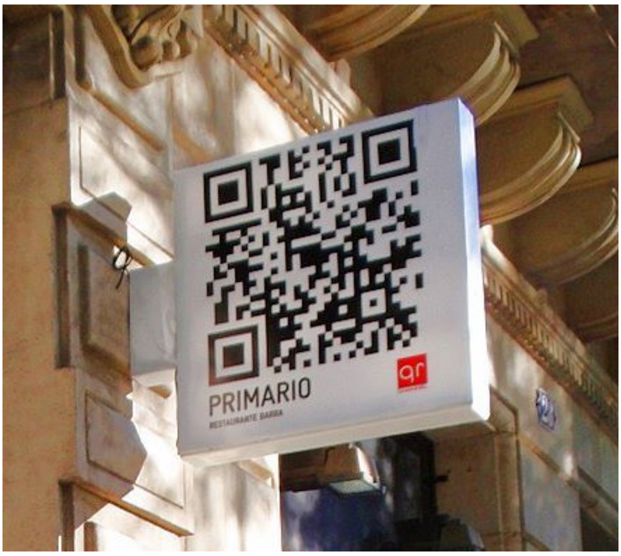
```
5000204892734
```

BarcodeRecognize[



] |

BarcodeRecognize[



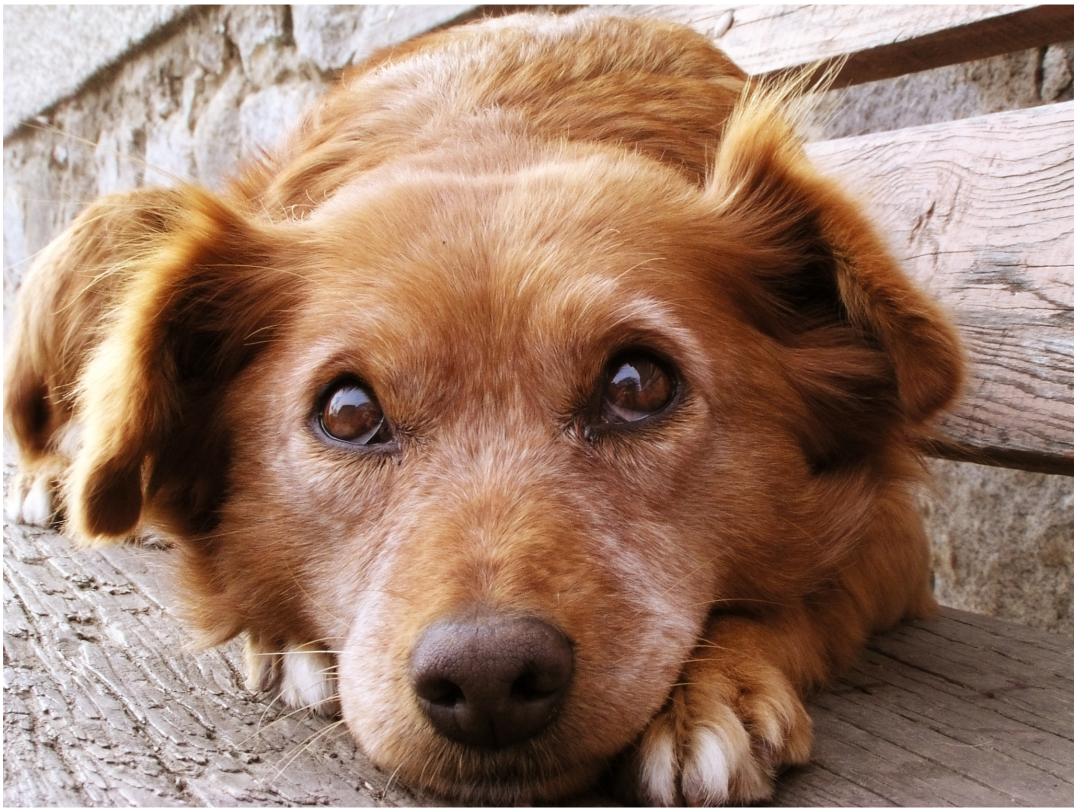
]

```
http://www.primario.com.es/qr
```

Reconhecimento de imagem

Image Identify: www.imageidentify.com

```
img1 = Import[
  "https://www.petfinder.com/wp-content/uploads/2012/11/dog-how-to-select-your-
    new-best-friend-thinkstock99062463-253x190.jpg"]
```



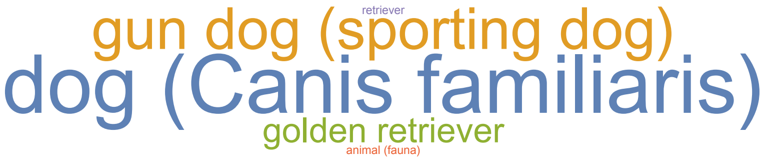
```
ImageIdentify[img1]
```

```
gun dog (sporting dog)
```

```
Table[ImageIdentify[img1, SpecificityGoal → i], {i, 0, 1, .1}]
```

```
{animal (fauna), dog (Canis familiaris), dog (Canis familiaris), dog (Canis familiaris),
 dog (Canis familiaris), gun dog (sporting dog), gun dog (sporting dog),
 gun dog (sporting dog), retriever, golden retriever, golden retriever}
```

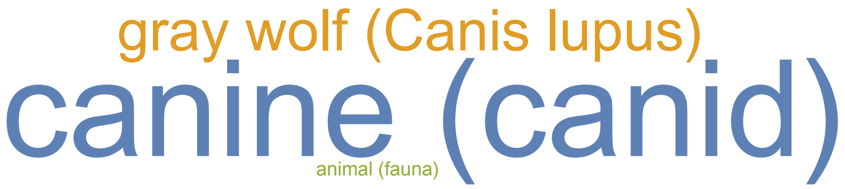
```
WordCloud[%]
```



```
img2 = Import[
  "https://www.imageidentify.com/public/prd/result/1/0/2/5/j/f/h/1/1/l/6/s/e/
    preview.jpeg?v=1.9&t=1431531931"]
```

```
Table[ImageIdentify[img2, SpecificityGoal → i], {i, 0, 1, .1}]
WordCloud[%]
```

```
{ animal (fauna) , canine (canid) , canine (canid) , canine (canid) ,
  canine (canid) , canine (canid) , canine (canid) , canine (canid) ,
  gray wolf (Canis lupus) , gray wolf (Canis lupus) , gray wolf (Canis lupus) }
```

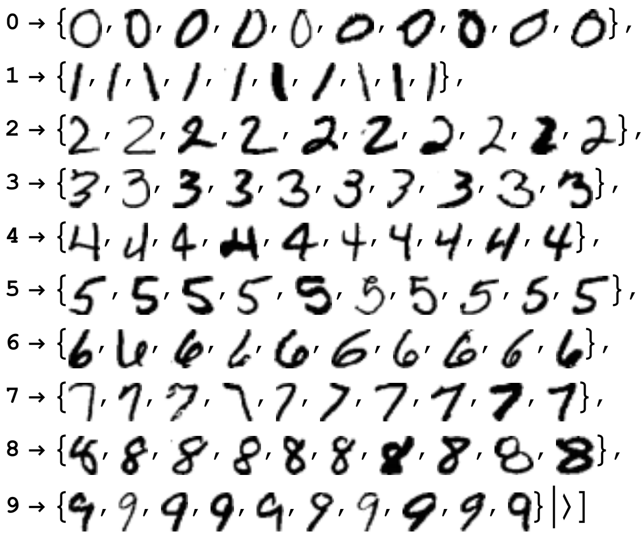


Classificação - Inteligência Artificial & Estatística

Fonte: Exemplos Workshop Wolfram

Classificação de dígitos manuscritos. Utilizando MNIST database banco de dados de digitos manuscritos

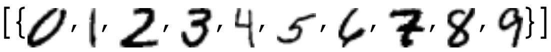
```
digit = Classify[⟨|
```



```
|⟩]
```

```
ClassifierFunction[ Method: LogisticRegression
                    Number of classes: 10 ]
```

```
digit[
```



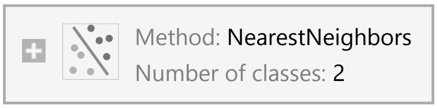
```
]
{0, 1, 2, 3, 4, 1, 4, 7, 8, 9}
```

```
digit[{0, 1, 2, 3, 4, 5, 6, 7, 8, 9}, "Probabilities"]
```

```
{<|0 → 0.684505, 1 → 0.0154666, 2 → 0.0153976, 3 → 0.0169518, 4 → 0.160923,
  5 → 0.0401903, 6 → 0.0400469, 7 → 0.00298294, 8 → 0.00156006, 9 → 0.0219758|>,
 <|0 → 0.00378951, 1 → 0.799247, 2 → 0.0589715, 3 → 0.0000618903, 4 → 0.000746523,
  5 → 0.0205328, 6 → 0.0684764, 7 → 0.000207845, 8 → 0.0413316, 9 → 0.00663466|>,
 <|0 → 0.000171339, 1 → 0.00469834, 2 → 0.976221, 3 → 0.0114145,
  4 → 5.47242 × 10^-6, 5 → 0.000421269, 6 → 0.00241323,
  7 → 0.00333904, 8 → 0.000336282, 9 → 0.000979774|>,
 <|0 → 0.0672796, 1 → 0.00964697, 2 → 0.030956, 3 → 0.735438, 4 → 0.000536483,
  5 → 0.0204429, 6 → 0.0138397, 7 → 0.0169709, 8 → 0.0539075, 9 → 0.0509818|>,
 <|0 → 0.00252346, 1 → 0.0113363, 2 → 0.000369817, 3 → 0.0332141, 4 → 0.772958,
  5 → 0.026577, 6 → 0.000607053, 7 → 0.0834587, 8 → 0.00167021, 9 → 0.0672856|>,
 <|0 → 0.00801577, 1 → 0.540073, 2 → 0.0100605, 3 → 0.196213, 4 → 0.00464873,
  5 → 0.031362, 6 → 0.203498, 7 → 0.00302653, 8 → 0.00294349, 9 → 0.000158906|>,
 <|0 → 0.242137, 1 → 0.00206954, 2 → 0.0261615, 3 → 0.00011534, 4 → 0.451554,
  5 → 0.01558, 6 → 0.255324, 7 → 0.000584, 8 → 0.000135414, 9 → 0.00633974|>,
 <|0 → 0.000289156, 1 → 0.0835541, 2 → 0.00167401, 3 → 0.0441853, 4 → 0.00292076,
  5 → 0.0349077, 6 → 0.000423702, 7 → 0.634737, 8 → 0.1587, 9 → 0.0386087|>,
 <|0 → 0.00359474, 1 → 0.0749362, 2 → 0.198509, 3 → 0.000794962, 4 → 0.0206894,
  5 → 0.00837472, 6 → 0.0413031, 7 → 0.0112217, 8 → 0.636507, 9 → 0.00406885|>,
 <|0 → 0.0103541, 1 → 0.0000248497, 2 → 5.09266 × 10^-6, 3 → 0.00130539,
  4 → 0.00858858, 5 → 0.00104905, 6 → 1.71033 × 10^-6,
  7 → 0.0003818, 8 → 0.000235849, 9 → 0.978054|>}
```

Classificação de imagens: Noite e Dia!!!

```
daynight =
 Classify[<|"Night" → {, , , , , , , , , , , , , , }, "Day" → {, , , ,
    , , , , , , , , , , }|>]
```



```
ClassifierFunction[ Method: NearestNeighbors
                    Number of classes: 2 ]
```

daynight[{ , ,

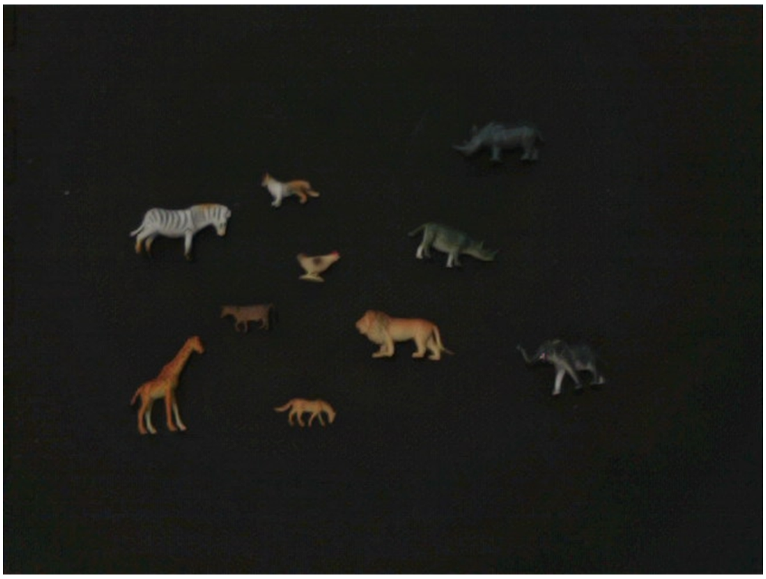
, , }]

```
{Day, Day, Night, Night, Day}
```

Contar objetos na imagem

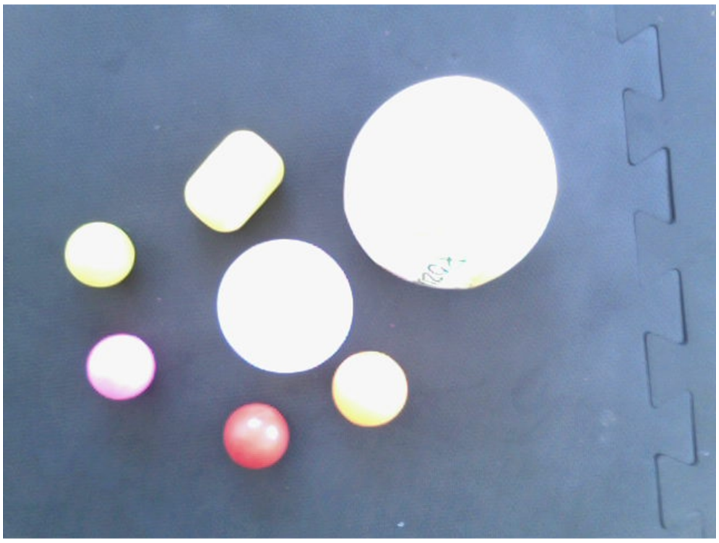
Imagens obtidas via Raspberry Pi

```
img3 =
```



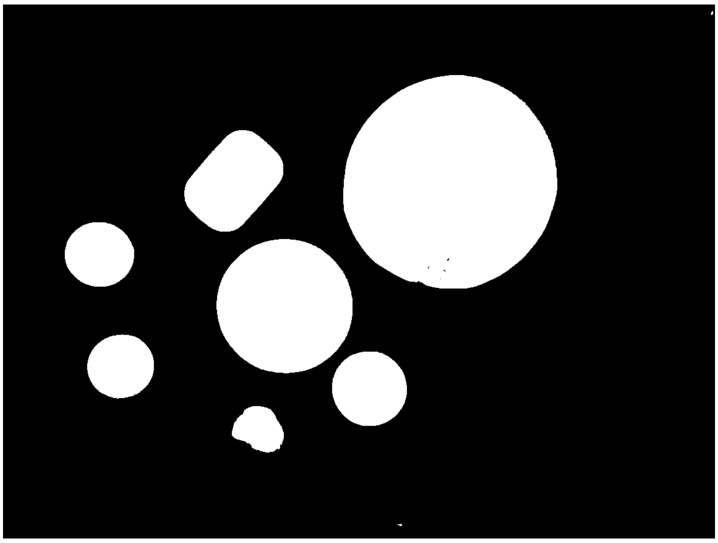
```
;
```

```
img3 =
```



```
;
```

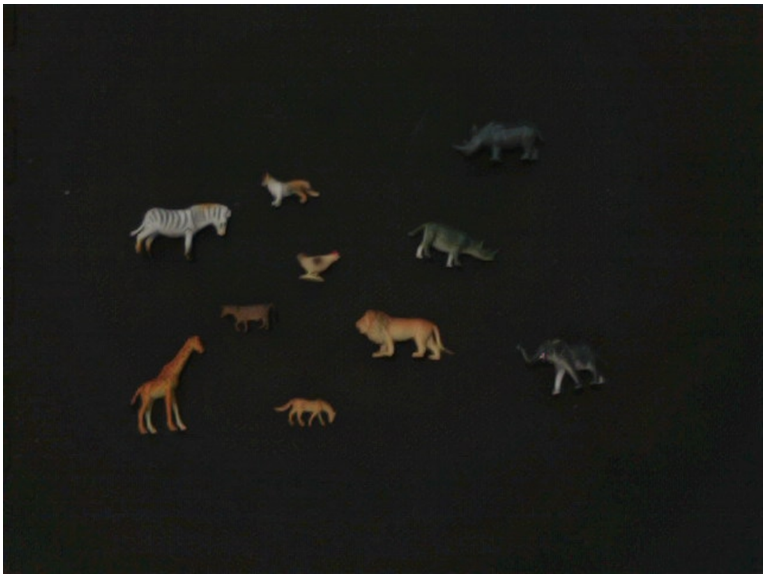
```
Binarize[img3]
```



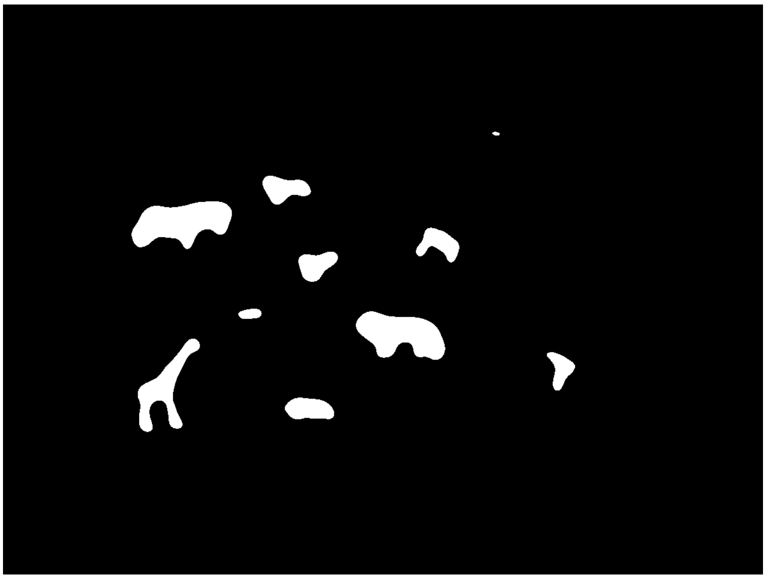
```
centroidData = ComponentMeasurements[
   Binarize[GaussianFilter[img3, 17]], {"Centroid"}][[All, 2, 1]]
```

```
{{724.972, 577.302}, {372.666, 579.039}, {155.168, 460.066},
 {455.68, 375.484}, {190.385, 279.429}, {592.861, 242.995}, {413.182, 174.423}}
```

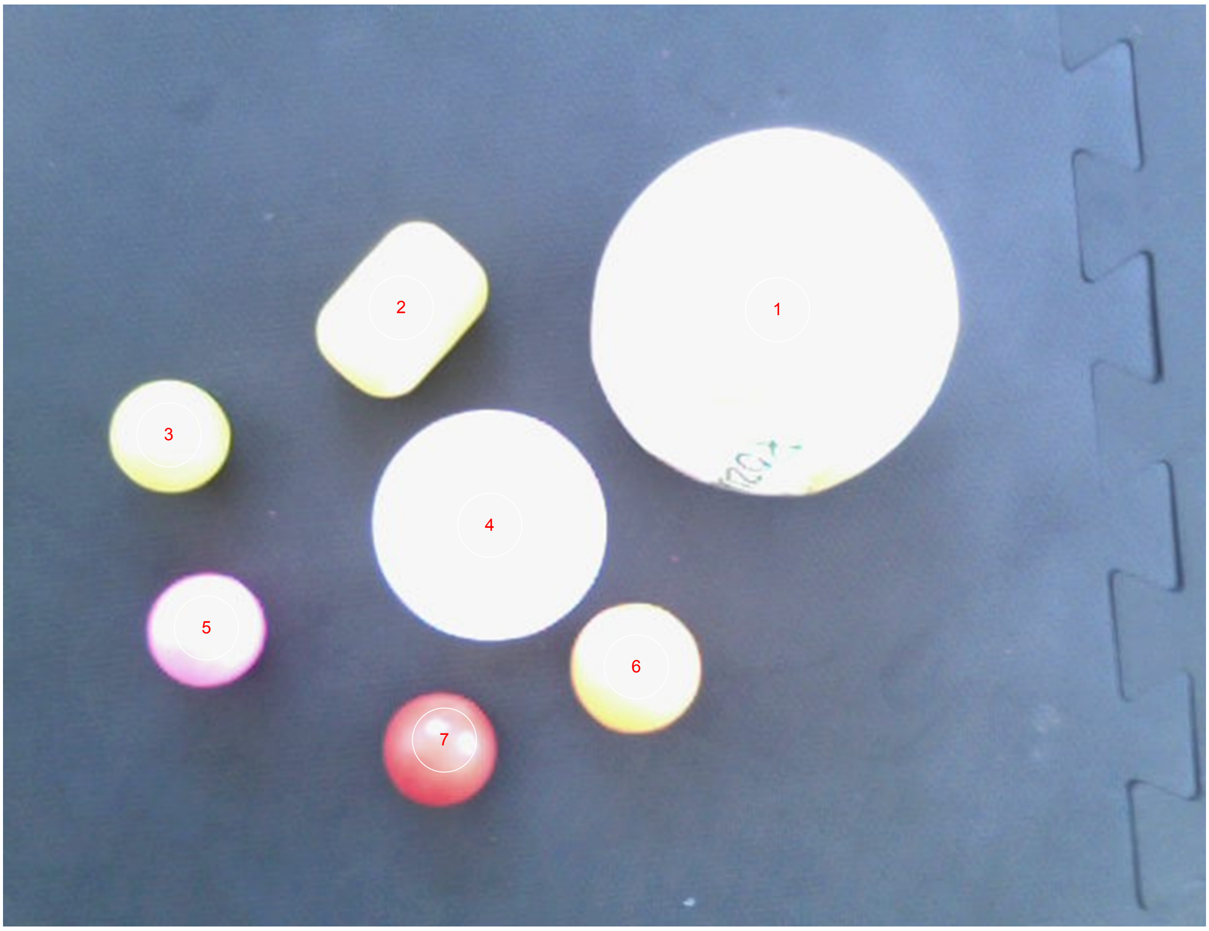
```
Binarize[
 GaussianFilter[
```



```
, 17]]
```



```
Show[Image[img3, ImageSize → 640],
 Graphics[{White,
   Circle[#, 30] & /@ centroidData,
   Red,
   Table[Inset[ToString[i], centroidData⟦i⟧],
    {i, 1, Length[centroidData]}]}]]
```



```
Binarize[Blur[img3, 20]]
```

```
RemoveBackground[img3, {"Background", Black}]
```

Automatos Celulares & DNA

Conceito: http://demonstrations.wolfram.com/CellularAutomataEvaluation/
Exemplo: http://demonstrations.wolfram.com/ApplyingTheSmithWatermanSimilarityToCellularAutomata/
Sequence/DNA Align: http://www.seas.gwu.edu/~simhaweb/cs151/lectures/module12/align.html

adenine, guanosine, thymine, cytosine

{ **adenine** (chemical) , **guanosine** (chemical) , **thymine** (chemical) , **cytosine** (chemical) }

{ adenine , guanosine , thymine , cytosine }

adenine »

3D structure

Assuming "adenine" is a chemical compound | Use as a word instead

Input interpretation:

adenine (chemical)

adenine

Chemical names and formulas:

More

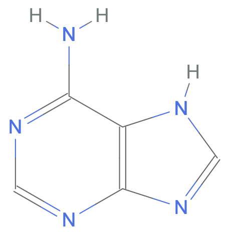
formula	$C_5H_5N_5$
name	adenine
IUPAC name	7H–purin–6–amine

Structure diagram:

Skeletal structure | ▾

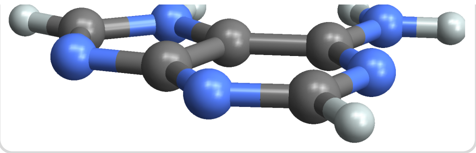
Show bond information

Step- by- step



3D structure:

Show space filling



Basic properties:

molar mass	135.127 g/mol
phase	solid (at STP)
melting point	360 °C
density	0.99172 g/cm^3
solubility in water	very soluble

+ Units

Hydrophobicity and permeability properties:

experimental LogP hydrophobicity	−0.3
predicted LogP hydrophobicity	−0.24
experimental LogS	−2.12
predicted LogS	−1.07

Basic drug properties:

More

approval status	approved | nutraceutical | small molecule
drug categories	dietary supplement | micronutrient

Solid properties (at STP):

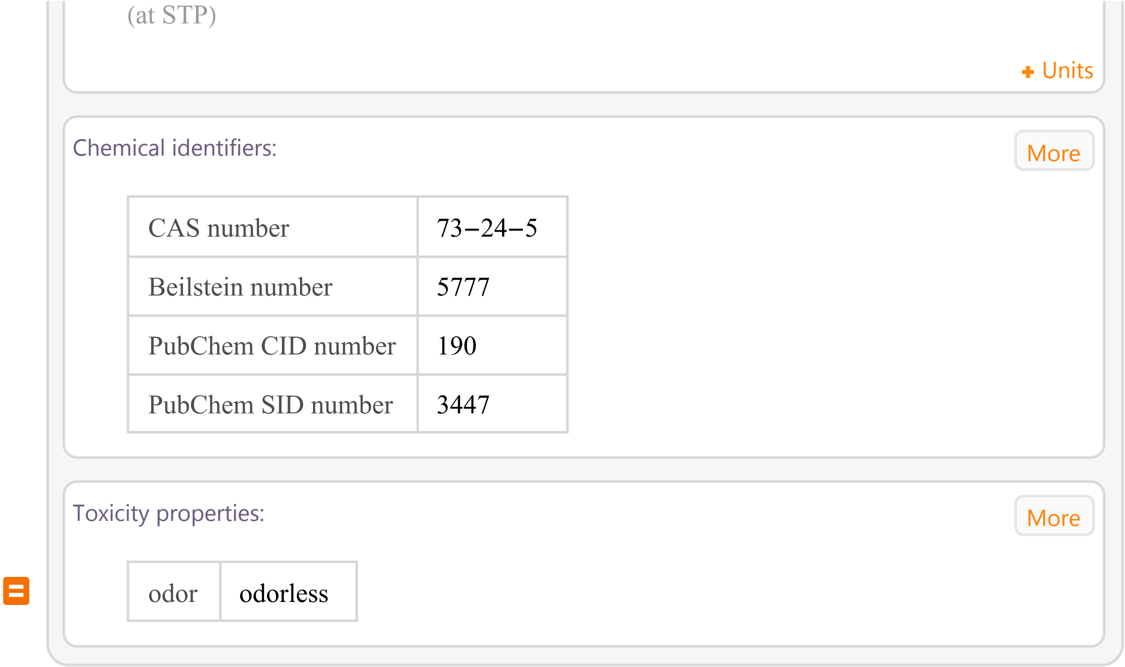
density	0.99172 g/cm^3
vapor pressure	0.033 mmHg (at 25 °C)

+ Units

Thermodynamic properties:

More

specific heat capacity c_p	solid	1.088 J/(g K)
specific heat of formation $\Delta_f H°$	gas	1.522 kJ/g
	solid	0.7171 kJ/g
specific heat of combustion	20.57 kJ/g	



(at STP)

Chemical identifiers:

CAS number	73–24–5
Beilstein number	5777
PubChem CID number	190
PubChem SID number	3447

Toxicity properties:

odor	odorless

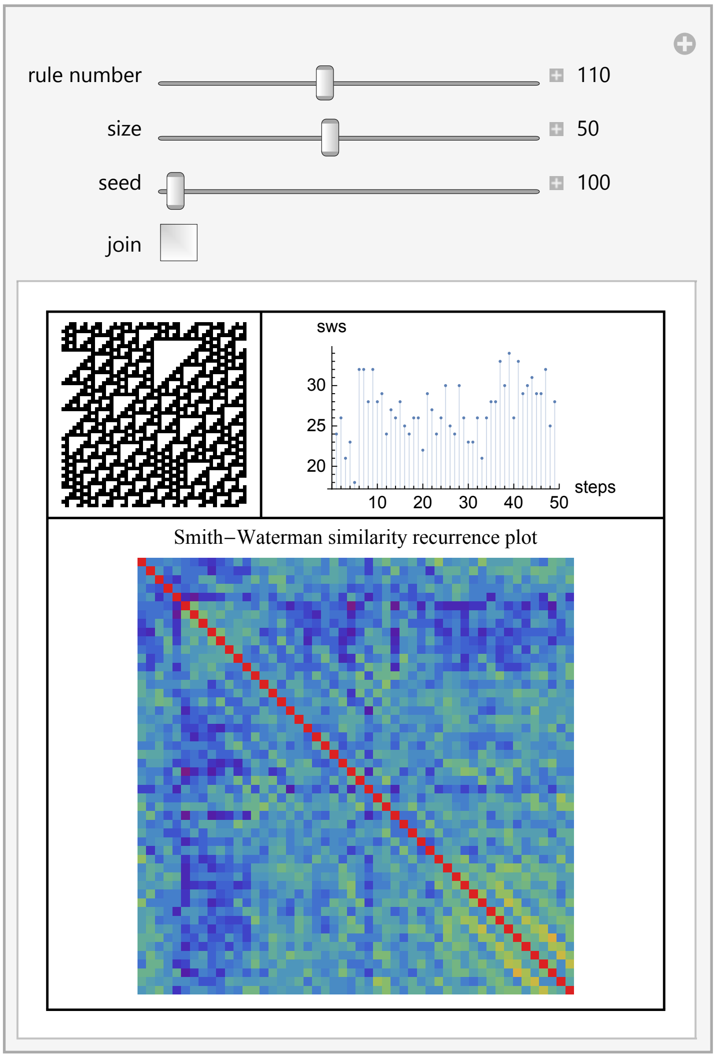
```
SequenceAlignment["GTCAA", "GTACC"]
```

```
{GT, {, A}, C, {AA, C}}
```

```
SmithWatermanSimilarity[{1, 0, 1}, {1, 1, 1}]
```

```
1.
```

```
Manipulate[RandomSeed[seed];
 Module[{ca, recurrence, swsSerie},
  ca = CellularAutomaton[rule, RandomInteger[1, size], size - 1];
  recurrence =
   Table[SmithWatermanSimilarity[xStep, yStep], {xStep, ca}, {yStep, ca}];
  swsSerie = SmithWatermanSimilarity[#〚1〛, #〚2〛] & /@
    Partition[Riffle[Drop[ca, -1], Rest[ca]], 2];

  Grid[{{
      ArrayPlot[ca, Frame → None, ImageSize → {100, 100}],
      ListPlot[swsSerie, Filling → Axis,
       AxesLabel → {"steps", "sws"}, Joined → join, ImageSize → {200, 100}] },
     {ArrayPlot[recurrence, Frame → None, ColorFunction → "Rainbow",
       ImageSize → {250, 250},
       PlotLabel → Text@"Smith-Waterman similarity recurrence plot"],
      SpanFromLeft}}, Frame → All, Alignment → {Center, Center}]],

 {{rule, 110, "rule number"}, 0, 255, 1, Appearance → "Labeled"},
 {{size, 50}, 10, 100, 1, Appearance → "Labeled"},
 {seed, 100, 10 000, 1, Appearance → "Labeled"},
 {join, {False, True}}
]
```



rule number 110

size 50

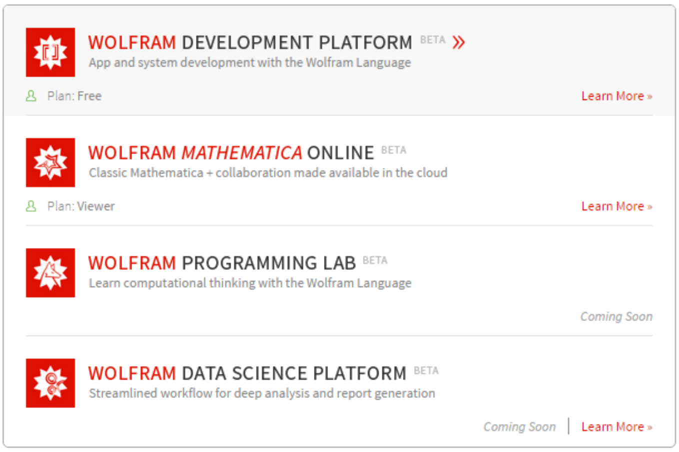
seed 100

join □

GEO + Serviço WEB Wolfram Cloud

Mathematica on-line: serviços para aplicações móveis e WEB

Wolfram Cloud: http://www.wolframcloud.com/



WOLFRAM DEVELOPMENT PLATFORM BETA »
App and system development with the Wolfram Language
Plan: Free — Learn More »

WOLFRAM *MATHEMATICA* ONLINE BETA
Classic Mathematica + collaboration made available in the cloud
Plan: Viewer — Learn More »

WOLFRAM PROGRAMMING LAB BETA
Learn computational thinking with the Wolfram Language
Coming Soon

WOLFRAM DATA SCIENCE PLATFORM BETA
Streamlined workflow for deep analysis and report generation
Coming Soon | Learn More »

Função **GetStreeMap** gera mapa com base na posição latitude e longitude

```
GetStreetMap[point_List] :=
  GeoGraphics[GeoMarker[point], GeoRange → Quantity[1, "Kilometer"]];
```

Função **CloudDeploy** e **APIFunction** combinadas geram serviço disponível na WEB (REST e JSON)

```
CloudDeploy[APIFunction[{"latitude" → "Number", "longitude" → "Number"},
  GeoGraphics[GeoMarker[{#latitude, #longitude}],
    GeoRange → Quantity[1, "Kilometer"]] &, "PNG"], Permissions → "Public"]
```

```
CloudObject[
 "https://www.wolframcloud.com/objects/faf09043-f977-4d02-b1e5-c132ff8e628a"]
```

Passando os parâmetros via URL o mapa é gerado, pode ser chamado de aplicação WEB ou Mobile via JavaScript ou outras linguagens como Java ou Python.

URL: https://www.wolframcloud.com/objects/faf09043-f977-4d02-b1e5-c132ff8e628a?latitude=-23.5417&longitude=-40.5650

REPÚBLICA
SÉ
Praça da Sé
Túnel do Anhangabaú
Rua Líbero Badaró
Rua Boa Vista
Viaduto Diário Popular
Avenida do Estado
Avenida do Estado - Pista Central
Rua da Figueira
Rua Riachuelo
Rua Tabatinguera
Avenida da Liberdade
Avenida Vinte e Três de Maio
Ligação Leste-Oeste
Rua dos Estudantes
Rua Doutor Lund
Rua São Paulo
Rua Barão de Iguape
Rua Leopoldo Miguez
Rua Santo Amaro
Rua Genebra
Rua Jaceguai
Rua Abolição
Rua Humaitá
Rua Major Diogo
Rua da Glória
Rua Dom Pedro II
Rua 7 de Abril
Avenida São Luís
Rua Maria Domitila
Rua do Glicério
Rua Otto de Alencar

Demo em JavaScript

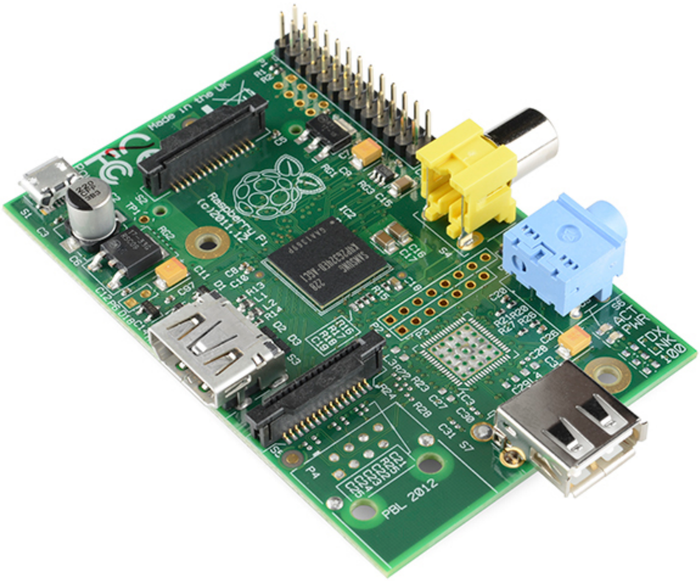
Internet das Coisas

Wolfram Language & *Mathematica* disponível em dispositivos

Raspberry Pi

Wolfram Language & Raspberry Pi

http://www.wolfram.com/raspberry-pi/



- Intel Edison

  Wolfram Language & Intel Edison

  http://www.wolfram.com/intel-edison/?source=frontpage