Project : Google Dino Game of - Automation

Google Chrome Dino Bot using Image Recognition

```
 Develop a Python-based project with NumPy , Pyautogui and PIL (Python Imaging Library) & Time Libraries for
implementation.
 This project is very basic and consists of only few lines of code.
```

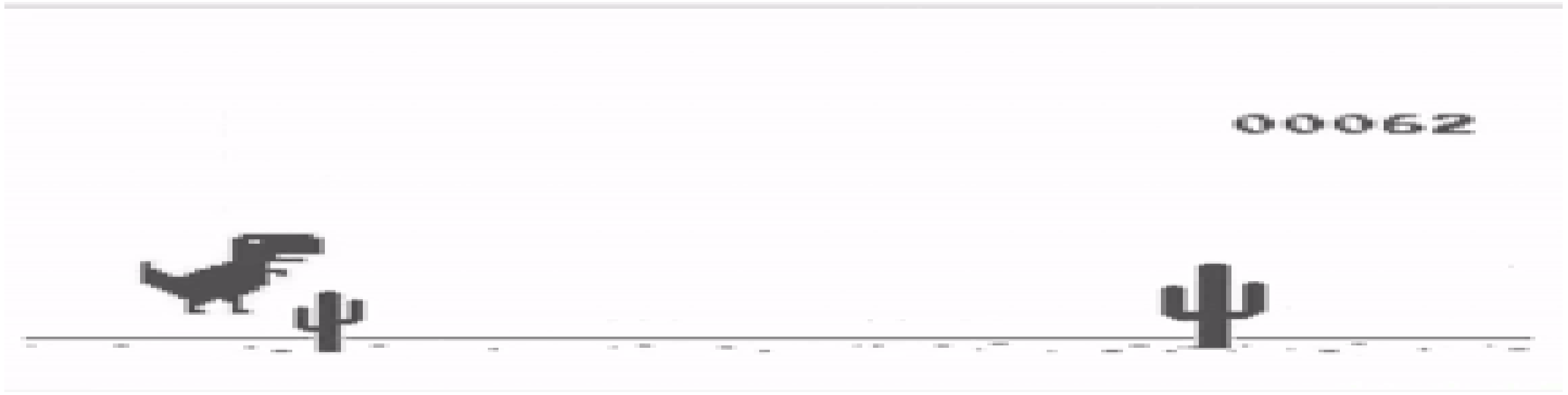

Table of Contents

Problem Statement:

- Click on the restart button using Pyautogui library using “replaybutton” coordinates.
- Calculate the sum of all white pixels values present in the box in front of Dinosaur.
- If the sum of pixels values present at any time in the box becomes less than the sum of white pixels values, it means either “bush” or “bird” is coming. So either we have to make our Dino jump or bend down.
- In order to protect Dino from “Bush”, we make a jump.

- In order to protect Dino from “Bird”, we always keep our Dino down.

---

1. Introduction

`We will work with NumPy , Pyautogui and PIL (Python Imaging Library) for implementation. This project is very basic and consists of only about 50 lines of code but its result will make you surprise.`

Some libraries used are:

PIL: Python Imaging Library (PIL) is a free library for the Python programming language that adds support for opening, manipulating, and saving many different image file formats.

Pyautogui: PyAutoGUI is a Python module for programmatically controlling the mouse and keyboard without any user interaction.

Time: Python “Time” Module which allows us to handle various operations regarding time, its conversions and representations, which find its use in various applications in life.

Numpy: NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays.

2. Algorithm

1. Click on the restart button using Pyautogui library using “replaybutton” coordinates.
2. Calculate the sum of all white pixels values present in the box in front of Dinosaur.
3. If the sum of pixels values present at any time in the box becomes less than the sum of white pixels values, it means either “bush” or “bird” is coming. So either we have to make our Dino jump or bend down.
4. In order to protect Dino from “Bush”, we make a jump.
5. In order to protect Dino from “Bird”, we always keep our Dino down.

   `Reference  click here.`

3. Python implementation:

Importing Libraries

In [ ]:

```
# importing above defined libraries to
from PIL import ImageGrab, ImageOps
import pyautogui
import time
import numpy as np
```

Python Class implementation:

In [ ]:

```
class coordinates():

    # coordinates of replay button to start the game
    replaybutton =(360, 214)
    # this coordinates represent the top-right coordinates
    # that will be used to define the front box
    dinasaur = (149, 239 )
```

Python Functions implementation:

In [ ]:

```
def restartGame():

    # using pyautogui library, we are clicking on the
    # replay button without any user interaction
    pyautogui.click(coordinates.replaybutton)

    # we will keep our Bot always down that
    # will prevent him to get hit by bird
    pyautogui.keyDown('down')
```

In [ ]:

```
def press_space():

    # releasing the Down Key
    pyautogui.keyUp('down')

    # pressing Space to overcome Bush
    pyautogui.keyDown('space')

    # so that Space Key will be recognized easily
    time.sleep(0.05)

    # printing the "Jump" statement on the
    # terminal to see the current output
    print("jump")
    time.sleep(0.10)
    # releasing the Space Key
    pyautogui.keyUp('space')

    # again pressing the Down Key to keep my Bot always down
    pyautogui.keyDown('down')
```

In [ ]:

```
def imageGrab():
    # defining the coordinates of box in front of dinosaur
    box = (coordinates.dinasaur[0]+30, coordinates.dinasaur[1],
           coordinates.dinasaur[0]+120, coordinates.dinasaur[1]+2)

    # grabbing all the pixels values in form of RGB tuples
    image = ImageGrab.grab(box)

    # converting RGB to Grayscale to
    # make processing easy and result faster
    grayImage = ImageOps.grayscale(image)

    # using numpy to get sum of all grayscale pixels
    a = np.array(grayImage.getcolors())

    # returning the sum
    print(a.sum())
    return a.sum()
```

In [ ]:

```
# function to restart the game
restartGame()
while True:
     # 435 is the sum of white pixels values of box.
     # You may get different value is you are taking bigger
     # or smaller box than the box taken in this article.
     # if value returned by "imageGrab" function is not equal to 435,
     # it means either bird or bush is coming towards dinosaur
     if(imageGrab()!= 435):
        press_space()
        # time to recognize the operation performed by above function
        time.sleep(0.1)
```

Below is the Complete Python implementation:

In [ ]:

```
# importing above defined libraries to
# implement the functionalities
from PIL import ImageGrab, ImageOps
import pyautogui
import time
import numpy as np

time.sleep(5)

class coordinates():

    # coordinates of replay button to start the game
    replaybutton =(360, 214)
    # this coordinates represent the top-right coordinates
    # that will be used to define the front box
    dinasaur = (149, 239 )

def restartGame():

    # using pyautogui library, we are clicking on the
    # replay button without any user interaction
    pyautogui.click(coordinates.replaybutton)

    # we will keep our Bot always down that
    # will prevent him to get hit by bird
    pyautogui.keyDown('down')

def press_space():

    # releasing the Down Key
    pyautogui.keyUp('down')

    # pressing Space to overcome Bush
    pyautogui.keyDown('space')

    # so that Space Key will be recognized easily
    time.sleep(0.05)

    # printing the "Jump" statement on the
    # terminal to see the current output
    print("jump")
    time.sleep(0.10)
    # releasing the Space Key
    pyautogui.keyUp('space')

    # again pressing the Down Key to keep my Bot always down
    pyautogui.keyDown('down')

def imageGrab():
    # defining the coordinates of box in front of dinosaur
    box = (coordinates.dinasaur[0]+30, coordinates.dinasaur[1],
           coordinates.dinasaur[0]+120, coordinates.dinasaur[1]+2)

    # grabbing all the pixels values in form of RGB tuples
    image = ImageGrab.grab(box)

    # converting RGB to Grayscale to
    # make processing easy and result faster
    grayImage = ImageOps.grayscale(image)

    # using numpy to get sum of all grayscale pixels
    a = np.array(grayImage.getcolors())

    # returning the sum
    print(a.sum())
    return a.sum()



# function to restart the game
restartGame()
while True:
     # 435 is the sum of white pixels values of box.
     # You may get different value is you are taking bigger
     # or smaller box than the box taken in this article.
     # if value returned by "imageGrab" function is not equal to 435,
     # it means either bird or bush is coming towards dinosaur
     if(imageGrab()!= 435):
        press_space()
        # time to recognize the operation performed by above function
        time.sleep(0.1)
```

4. Improvements :

Over a period of time, the Dino Bot Game becomes fast. The Birds and Bushes start coming very fast. So we are not making our Bot to learn all these things, changing its speed based on past learning. So our bot will function for around 2000 score. In order to score more, we have to apply machine learning and artificial intelligence.

Google Chrome Dino Bot using Image Recognition
Notepad Link