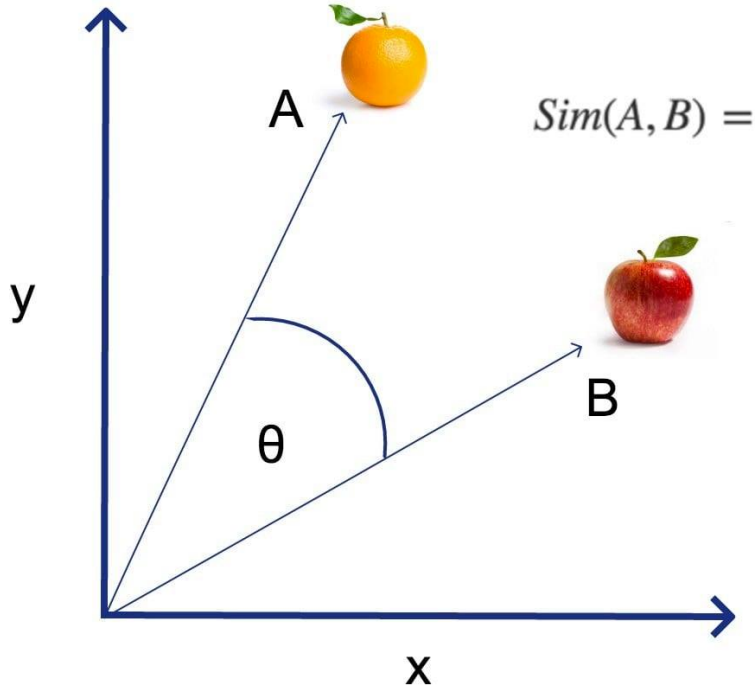
OBJECT TRACKING

Cosine Similarity



$$Sim(A,B) = \cos(\theta) = \frac{A \cdot B}{\|A\| \|B\|}$$

Huỳnh Tấn Khải

10.11.2022

PROGRAMMING 1

PROF. VINH

INTRODUCTION (Problem)

Given a template (a small image), we are supposed to track that image in 63 bigger images where the object (my professor) gradually changes.

HYPOTHESIS

We can calculate the cosine similarity between the template image and the portions of the bigger image. The portion which corresponds to the largest cosine value (closest to 1) will match the most with the template, and thus the template could be detected.

MATERIALS

1. C Programming
2. An IDE for C
3. Predefined libraries to read, load, and write an image

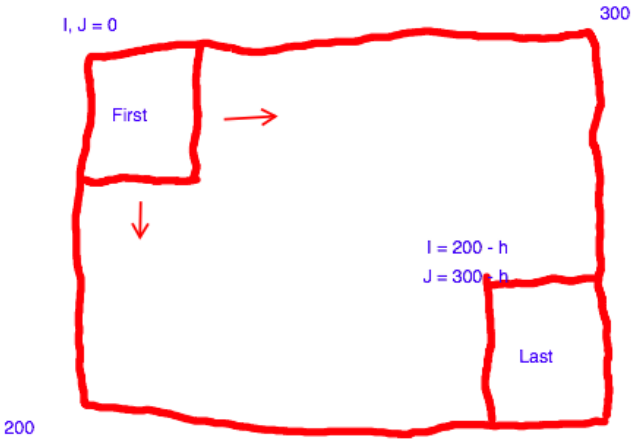
PROCEDURE

1. Convert template and all samples to grayscale

This method will convert the pixels of the image from 3 to 1 channel only.

2. Computing cosine similarity while sliding the template through the sample

- Sliding the template till the end of the image



```
// initialize
int h_limit = 400-h;
int w_limit = 530-w;

float cosine[h_limit * w_limit];

float max = cosine[0];
for(int a=88; a<h_limit; a++)
{
    for(int b=225; b<w_limit; b++)
    {
        float dot_product = 0.0;
        float template_len = 0.0;
        float sample_len = 0.0;
        for(int i=0, I=a;  i<h  ; i++, I++)
        {
            for(int j=0, J=b;  j<w  ; j++, J++)
```

To cut down the processing time, I slide the template only in the area where the object changes: height (88→400) and (225→530).

- Computing the cosine similarity while sliding

$$\cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \sqrt{\sum_{i=1}^{n} B_i^2}}$$

```
cosine[a*w_limit+b] = dot_product / (sqrt(template_len) * sqrt(sample_len));
```

- Drawing a rectangular border (the color of your choice) around the portion with greatest cosine value

```
// draw
if(I == a || I == a + h - 1 || J == b || J == b + w - 1)
{
    // sample_grey[I * W + J] = 0;
    sample[I * W * 3 + J * 3 + 0] = 0;
    sample[I * W * 3 + J * 3 + 1] = 0;
    sample[I * W * 3 + J * 3 + 2] = 0;
}
```

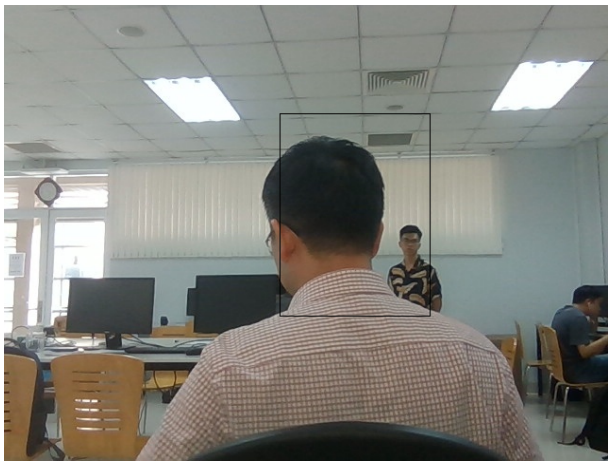
3. Updating the template for better accuracy

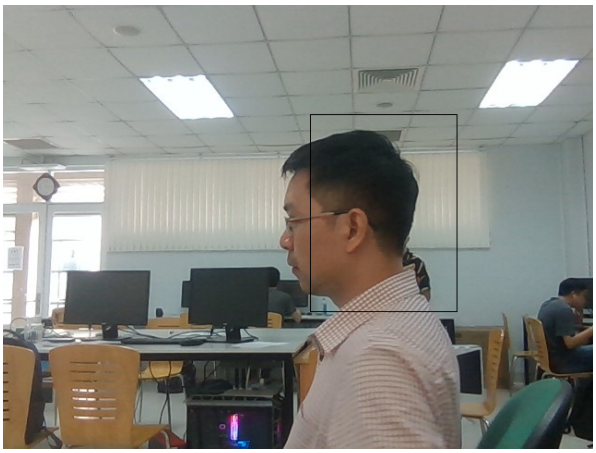
As we successfully detect the object in the sample, we update it to the template for the object tracking in the next sample.

The updated template will be more suited for the next evaluation.

RESULTS

These are 6 outputs for reference.





CONCLUSION

★ The cosine similarity algorithm when applied is very useful for the implementation of object tracking in C.

★ Although the coding process is pretty long, its pattern is not too complicated as long as we know how to slide the template object through the sample.

★ The main disadvantage of this implementation is that the cosine tracking may get less accurate as the object slightly moves in the sample.

REFERENCES

1. https://tvd12.com/cosine-similarity/

2. Source code: https://drive.google.com/drive/folders/1lDCbz9n925hnzX_9SO4CGpOGmT_HmTOW

3. Demo: https://www.youtube.com/watch?v=GhWpkTBlxb4