

ĐẠI HỌC QUỐC GIA TP.HCM
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN



KHOA: KHOA HỌC MÁY TÍNH

MÔN: NHẬP MÔN THỊ GIÁC MÁY TÍNH - CS231.N21.KHTN

Single Object Tracking

Thành viên:
Trương Thanh Minh - 21520064

Giảng viên:
TS. Mai Tiến Dũng

Mục lục

1	Bài toán	2
1.1	Single Object Tracking (SOT) là gì?	2
1.2	Input, Output	2
1.2.1	Input	2
1.2.2	Output	3
1.3	Khó khăn, thách thức của bài toán	4
2	Lý do chọn đề tài	6
3	Phương pháp	7
3.1	Giới thiệu một số phương pháp	7
3.2	Phương pháp dùng trong đề tài này	8
3.3	Input, Output của phương pháp được dùng	9
3.4	So sánh 2 phương pháp	9
4	Thực nghiệm	10
4.1	Dataset	10
4.2	Dộ đo đánh giá	12
5	Kết quả	12
5.1	Kết quả khi sử dụng phương pháp CSRT	12
5.2	Kết quả khi sử dụng phương pháp KCF	13
6	Demo	14
6.1	Demo trên các chuỗi khung hình trong bộ data OTB2015	14
6.2	Demo trên các video tự cung cấp	18
7	Các link liên quan tới báo cáo	19

1 Bài toán

1.1 Single Object Tracking (SOT) là gì?

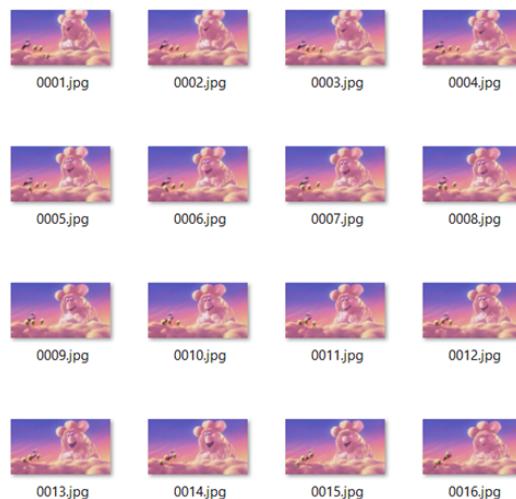
Single Object Tracking là quá trình xác định và theo dõi vị trí của một đối tượng di động trong một chuỗi hình ảnh hoặc video.

1.2 Input, Output

1.2.1 Input

Input của bài toán Single Object Tracking gồm 2 phần:

1. Chuỗi hình ảnh hoặc video mà trong đó chứa đối tượng ta muốn theo dõi

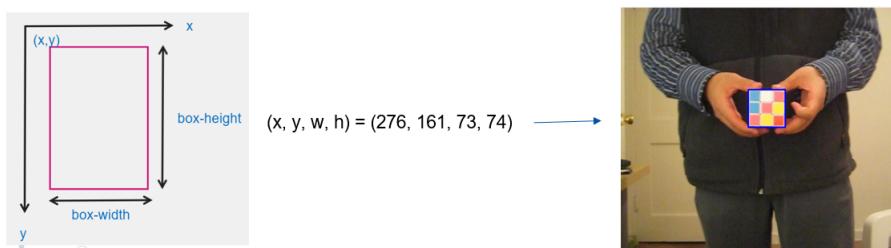


Hình 1: Chuỗi hình ảnh



Hình 2: Video

- Nếu đầu vào là chuỗi hình ảnh, tất cả các hình ảnh được đặt tên theo thứ tự các khung hình xuất hiện trong video.
2. **Vị trí ban đầu của đối tượng (trong khung hình đầu tiên) mà ta muốn theo dõi.**
- Vị trí ban đầu của đối tượng thường được xác định bằng cách xác định **bounding box (BB)** xung quanh đối tượng trong khung hình ban đầu.
 - BB sẽ gồm 4 giá trị:
 - x : Tọa độ theo trục x của góc trên trái trong khung hình đầu tiên.
 - y : Tọa độ theo trục y của góc trên trái trong khung hình đầu tiên
 - w : Chiều rộng của đối tượng mà mình muốn xác định
 - h : Chiều cao của đối tượng mà mình muốn xác định
 - Các giá trị trong BB có thể cách nhau bằng 1 khoảng cách hoặc dấu ','. Ví dụ các BB hợp lệ:
 - 1 2 3 4
 - 1,2,3,4



Hình 3: Ví dụ

Từ hình trên, ta có thể thấy rằng, khi có 1 BB, ta có thể dễ dàng xác định được chính xác vị trí của một đối tượng.

1.2.2 Output

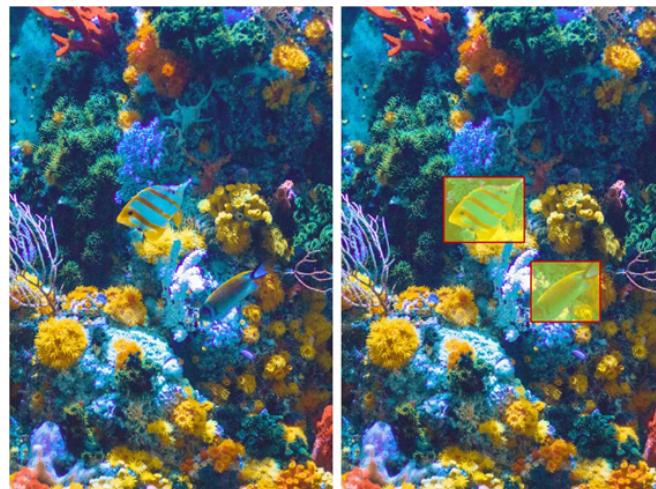
Xác định BB xung quanh đối tượng ở mỗi khung hình. Định dạng của BB (giống với BB của vị trí đầu tiên của object)

- x : Tọa độ theo trục x của góc trên trái trong khung hình đầu tiên.
- y : Tọa độ theo trục y của góc trên trái trong khung hình đầu tiên
- w : Chiều rộng của đối tượng mà mình muốn xác định
- h : Chiều cao của đối tượng mà mình muốn xác định

1.3 Khó khăn, thách thức của bài toán

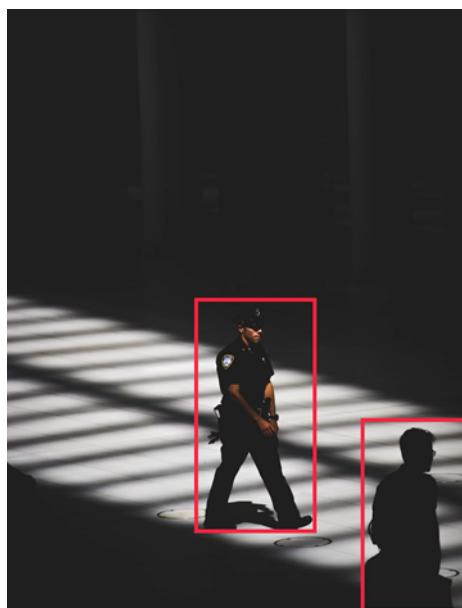
Bài toán Single Object Tracking được dùng khá nhiều trong cuộc sống hằng ngày, tuy nhiên bên cạnh đó vẫn tồn tại một số thách thức. Một trong số đó là:

1. **Bối cảnh lộn xộn (Background clutter)** Background có mật độ dày đặc, khi đó, khó để trích xuất các features, detect hoặc thậm chí tracking một đối tượng vì nó tạo ra nhiều thông tin dư thừa hoặc nhiễu. Từ đó gây ra việc khó để tiếp nhận được các đặc trưng quan trọng.



Hình 4: Background Clutters

2. **Biến đổi ánh sáng (Illumination Variation)** Độ sáng trên một đối tượng mà mình quan tâm thay đổi đáng kể khi đối tượng di chuyển. Khi đó việc định vị và ước tính của đối tượng trở nên khó khăn hơn.



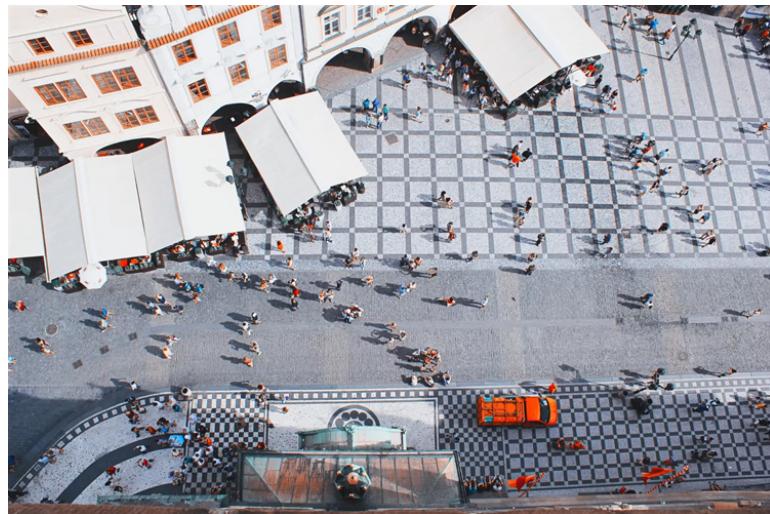
Hình 5: Illumination Variation

3. **Che khuất (Occlusion)** Khi đối tượng di chuyển qua các khu vực có nhiều vật thể khác hoặc có chuyển động gần đó, gây che khuất một phần hoặc toàn bộ đối tượng trong khung hình.



Hình 6: Occlusion

4. **Độ phân giải thấp (Low resolution)** Khi số pixel trong BB xung quanh đối tượng quá thấp. Điều đó sẽ làm ta khó theo dõi đối tượng mà mình quan tâm.



Hình 7: Low resolution

5. Biến đổi tỷ lệ (Scale Variation)

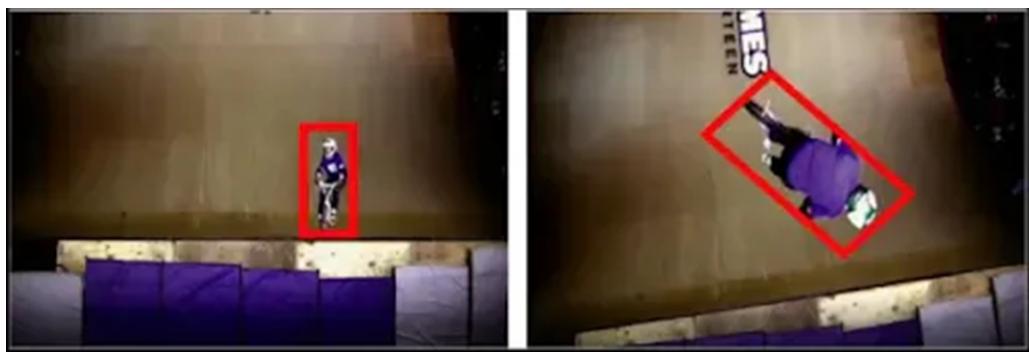
- Là sự thay đổi về tỷ lệ kích thước của đối tượng trong quá trình theo dõi. Điều này xảy ra khi đối tượng di chuyển gần hoặc xa khỏi camera, gây ra thay đổi về kích thước trên khung hình.

- Khi thay đổi kích thước, các đặc trưng hoặc hình dạng của nó cũng thay đổi → mất mát thông tin hoặc gây nhầm lẫn cho thuật toán tracking, khó khăn trong việc xác định vị trí chính xác và hình dạng của đối tượng.



Hình 8: Scale Variation

6. **Thay đổi hình dạng của đối tượng** Trên khung hình, hình dạng của đối tượng có thể xoay, giảm kích thước, biến dạng ...



Hình 9: Thay đổi hình dạng của đối tượng

7. **Chuyển động nhanh (Fast Motion)** Khi theo dõi đối tượng chuyển động nhanh, điều đó sẽ làm ảnh hưởng tới khả năng theo dõi chính xác đối tượng trên khung hình.

2 Lý do chọn đề tài

Có khá nhiều lý do để chọn bài toán Single Object Tracking cho đồ án này. Tuy nhiên, có 3 lý do chính dưới đây:

1. Ứng dụng thực tế

- Có thể được áp dụng trong nhiều lĩnh vực: giám sát an ninh, xe tự hành, quản lý giao thông, nhận dạng đối tượng và nhiều ứng dụng khác.
- Có thể đóng góp vào việc cải thiện hiệu suất và độ chính xác của các hệ thống thị giác máy tính thực tế.

2. Thách thức nghiên cứu

- Có nhiều yếu tố có thể ảnh hưởng đến quá trình tracking: thay đổi ánh sáng, độ phân giải, biến dạng hình dạng, che khuất và chuyển động nhanh → có thể mang lại sự đóng góp cho cộng đồng nghiên cứu và khám phá ra những khía cạnh mới trong việc xử lý và hiểu hình ảnh.

3. Tiềm năng phát triển

- Có thể góp phần tăng hiệu suất của một số hệ thống thị giác máy tính.
- Mở ra cơ hội đóng góp vào tiến bộ công nghệ và thúc đẩy sự phát triển của lĩnh vực này.

3 Phương pháp

3.1 Giới thiệu một số phương pháp

Có khá nhiều phương pháp để giải quyết bài toán Single Object Tracking (SOT). Dưới đây là một số phương pháp:

Tên thuật toán	Người phát minh	Năm phát minh
Kalman filter	Rudolf E. Kalman	1960
Particle filter	Leonard E. Baum và Donald G. Clifford	1960 - 1970
Mean shift	Dorin Comaniciu và Peter Meer	1999

Bảng 1: Một số thuật toán giải quyết bài toán SOT

Ngoài những phương pháp trên, còn có một số thuật toán khác được tích hợp trong thư viện OpenCV như:

Tên thuật toán	Paper	Năm
Boosting	Real-Time Tracking via On-line Boosting	2006
MIL (Multiple Instance Learning)	Visual tracking with online Multiple Instance Learning	2009
MedianFlow	Forward-Backward Error: Automatic Detection of Tracking Failures	2010
MOSSE (Minimum Output Sum of Squared Error)	Visual object tracking using adaptive correlation filters	2010
TLD (Tracking Learning Detection)	Tracking-Learning-Detection	2011
KCF (Kernelized Correlation Filter)	High-Speed Tracking with Kernelized Correlation Filters	2014
GOTURN (Generic Object Tracking Using Regression Networks)	Learning to Track at 100 FPS with Deep Regression Networks	2016
CSRT (Channel and Spatial Reliability Tracker)	Discriminative Correlation Filter Tracker with Channel and Spatial Reliability	2017

Bảng 2: Một số thuật toán giải quyết bài toán SOT được tích hợp trong OpenCV

3.2 Phương pháp dùng trong đề tài này

Có thể thấy, có khá nhiều thuật toán để giải quyết bài toán SOT. Trong những thuật toán được liệt kê ở trên, mình chọn thuật toán **KCF** và **CSRT**.

Lý do chọn 2 thuật toán này là vì:

- Cả 2 thuật toán này đều được tích hợp trong thư viện OpenCV. Mà OpenCV là thư viện rất là quen thuộc ở trong môn này.
- Chọn **CSRT** vì đây là thuật toán được tích hợp trong OpenCV mới nhất để giải quyết bài toán SOT (được đăng vào năm 2017).
- Chọn **KCF** vì đây là thuật toán có số lượng Citations trên Google Scholar khá cao. Điều này cho thấy đây là một thuật toán được khá nhiều người quan tâm.

1. KCF (Kernelized Correlation Filters)

- **Chi tiết:** [High-speed tracking with kernelized correlation filters](#).
- **Năm công bố:** 2014.
- **Tác giả:** João F. Henriques, Rui Caseiro, Pedro Martins, Jorge Batista.
- **Số lượng Citations trên Google Scholar** (tính đến 28/05/2023): 5973.

2. CSRT (Channel and Spatial Reliability Tracker)

- **Chi tiết:** [Discriminative Correlation Filter with Channel and Spatial Reliability](#).

- **Năm công bố:** 2017.
- **Tác giả:** Alan Lukezic, Tomas Vojir, Luka Cehovin Zajc, Jiri Matas, Matej Kristan.
- **Số lượt Citations trên Google Scholar**(tính đến 28/05/2023): 1306.

3.3 Input, Output của phương pháp được dùng

Về input, output của 2 phương pháp được tích hợp trong OpenCV này về cơ bản là khá giống nhau:

- **Input**

- **Khung hình hiện tại:** Chứa đối tượng cần được theo dõi.
- **ROI(Region of interest):** Đây là vùng chứa đối tượng cần theo dõi trong khung hình đầu tiên. Vùng này được xác định bằng 4 giá trị x, y, w, h với ý nghĩa như đã đề cập ở các phần trên.

- **Output**

- Vị trí **mới** của đối tượng trong khung hình hiện tại, được xác định thông qua 4 giá trị x, y, w, h .

3.4 So sánh 2 phương pháp

	KCF	CSRT
Tốc độ	Nhanh	Chậm
Độ chính xác	Ít chính xác	Chính xác hơn. Đặc biệt trong các tình huống như mất đối tượng tạm thời hoặc nhiễu
Hiệu suất khi bị che khuất	Hiệu suất không cao	Hiệu suất cao hơn
Độ phức tạp tính toán	Ít tính toán phức tạp	Tính toán phức tạp hơn
Tính tổng quát	Thường hoạt động tốt trong các tình huống đối tượng có chuyển động nhanh và không có sự thay đổi lớn về hình dạng hoặc kích thước	Có khả năng xử lý các TH đối tượng có sự biến đổi nhanh về hình dạng hoặc kích thước

Bảng 3: So sánh KCF với CSRT

Từ bảng trên, theo ý kiến cá nhân của mình, ta sử dụng:

- **KCF:** Khi cần **FPS cao**, nhưng có thể xử lý với **độ chính xác thấp** hơn một chút.
- **CSRT:** Khi cần **độ chính xác cao** hơn và có thể chịu được **FPS thấp**.

4 Thực nghiệm

4.1 Dataset

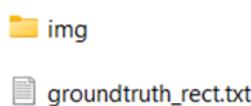
Có khá nhiều bộ dataset dùng trong bài toán Single Object Tracking. Một trong những bộ dataset là [Object Tracker Benchmark \(OTB2015\)](#) và đó cũng là bộ dataset mà mình dùng trong bài này.



Hình 10: Visualize chuỗi khung hình Bird2 trong bộ OTB2015

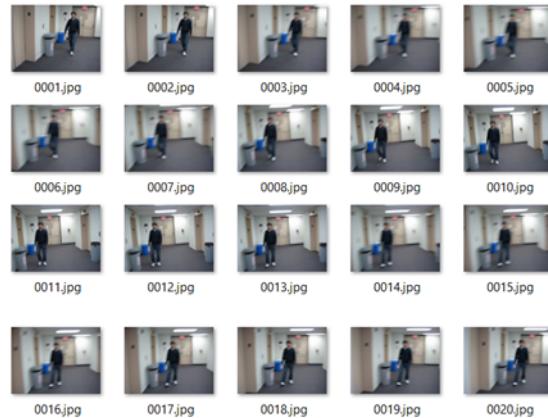
Về bộ data này:

- Chứa 98 chuỗi khung hình.
- Số lượng khung hình tối thiểu trong một chuỗi là 71 khung hình, tối đa là 3872 khung hình, trung bình số lượng khung hình là 590.
- Tổng số lượng khung hình là $59K$.
- Có độ phân giải khác nhau. Khung hình có độ phân giải thấp nhất là 128×96 và cao nhất là 800×336 .
- Một chuỗi sẽ gồm 2 thông tin bắt buộc:
 - img
 - groundtruth_rect.txt



Hình 11: Thông tin bắt buộc của một chuỗi khung hình

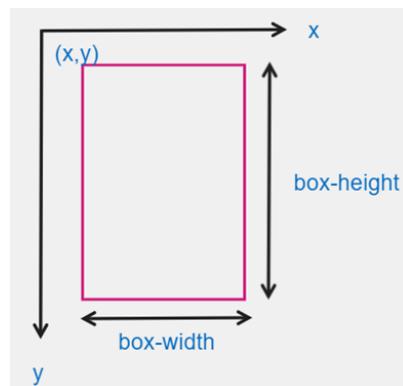
- Folder img sẽ gồm các khung hình được cắt ra từ video.



Hình 12: 20 khung hình đầu tiên trong chuỗi BlurBody

- Nội dung file groundtruth_rect.txt:

- Mỗi hàng sẽ đại diện cho BB của đối tượng được tracking tại frame đó ($x, y, box-width, box-height$).



Hình 13: Hình ảnh minh họa tọa độ

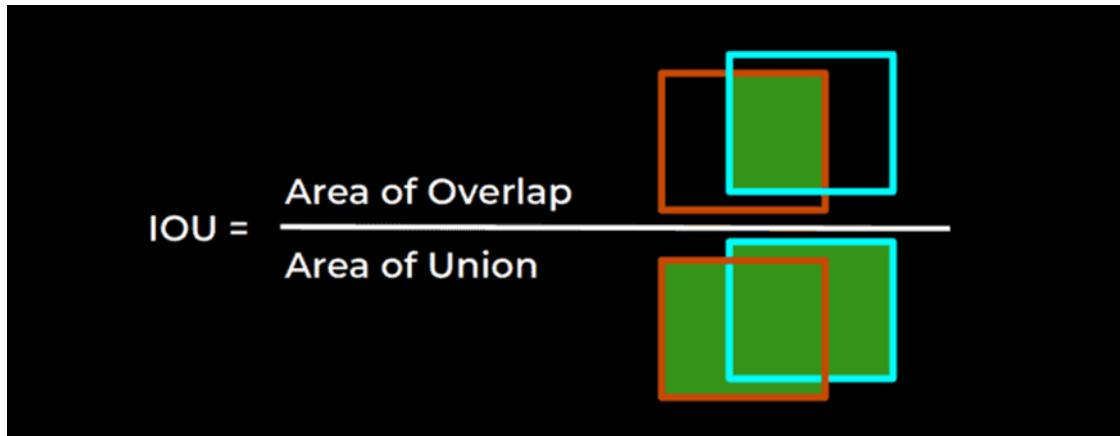
198,214,34,81
197,214,34,81
195,214,34,81
193,214,34,81
190,217,34,81
188,221,34,81
186,224,34,81
186,227,34,81
181,234,34,81
187,233,34,81
186,232,34,81

Hình 14: Ví dụ về nội dung file groundtruth_rect.txt

4.2 Độ đo đánh giá

Trong bài toán này, mình dùng **IoU (Intersection over Union)** để đánh giá hiệu quả của mô hình.

- IoU là thang đo được dùng để đánh giá sự trùng lặp giữa 2 vùng quan tâm. Trong bài toán này, IoU được dùng để đánh giá mức độ trùng lặp giữa **BB Predict** và **BB GroundTruth** xung quanh đối tượng mà mình muốn theo dõi (Tracking). Ta có thể tính IoU theo công thức như hình bên dưới.



Hình 15: Minh họa cách tính IoU

- Giá trị IoU nằm trong khoảng từ 0 đến 1. Trong đó:
 - 0: Không trùng lặp.
 - 1: Trùng lặp hoàn toàn.

5 Kết quả

5.1 Kết quả khi sử dụng phương pháp CSRT

Khi đánh giá trên bộ dataset OTB2015 bằng cách sử dụng thuật toán CSRT, ta thu được kết quả IoU trung bình trên toàn bộ dataset đó là **0.525**.

Cụ thể 10 chuỗi khung hình có giá trị IoU trung bình **cao nhất** là:

Tên chuỗi khung hình	Giá trị IoU trung bình
CarDark	0.872371
Coupon	0.861974
Car4	0.841898
Car2	0.83589
Man	0.821353
David2	0.821152
BlurCar2	0.820877
BlurOwl	0.811198
Dudek	0.805347
Single1	0.80006

Bảng 4: 10 chuỗi khung hình có giá trị IoU trung bình cao nhất khi sử dụng CSRT

Dưới đây là 10 chuỗi khung hình có giá trị IoU trung bình **thấp nhất**:

Tên chuỗi khung hình	Giá trị IoU trung bình
Human3	0.015626
Singer2	0.032043
Jump	0.041257
Skiing	0.049268
Girl2	0.05329
MotorRolling	0.08038
Ironman	0.083284
Matrix	0.084626
Freeman4	0.127691
DragonBaby	0.178138

Bảng 5: 10 chuỗi khung hình có giá trị IoU trung bình thấp nhất khi sử dụng CSRT

5.2 Kết quả khi sử dụng phương pháp KCF

Khi đánh giá trên bộ dataset OTB2015 bằng cách sử dụng thuật toán KCF, ta thu được kết quả IoU trung bình trên toàn bộ dataset đó là **0.189**.

Cụ thể 10 chuỗi khung hình có giá trị IoU trung bình **cao nhất** là:

Tên chuỗi khung hình	Giá trị IoU trung bình
FaceOcc1	0.760306
BlurCar4	0.740109
Dudek	0.739388
FaceOcc2	0.728963
Mhyang	0.693944
Girl2	0.668255
Board	0.623309
Car2	0.615697
Liquor	0.609912
FleetFace	0.577652

Bảng 6: 10 chuỗi khung hình có giá trị IoU trung bình cao nhất khi sử dụng KCF

Dưới đây là 10 chuỗi khung hình có giá trị IoU trung bình **thấp nhất**:

Tên chuỗi khung hình	Giá trị IoU trung bình
Human3	0.002577
Bird1	0.003095
Basketball	0.003718
Bolt	0.004285
Shaking	0.004806
Bolt2	0.005204
Matrix	0.005319
Human6	0.006286
Crowds	0.007625
Deer	0.008437

Bảng 7: 10 chuỗi khung hình có giá trị IoU trung bình thấp nhất khi sử dụng KCF

6 Demo

6.1 Demo trên các chuỗi khung hình trong bộ data OTB2015

Trong phần này, mình chỉ show ra demo của các chuỗi khung hình có giá trị trung bình IoU cao nhất và thấp nhất ở thuật toán CSRT và KCF. Nếu có nhu cầu demo trên các chuỗi khung hình khác, có thể lấy code trên [link github](#) mình để ở bên dưới để chạy.

1. CarDark (IoU cao nhất của thuật toán CSRT)



Hình 16: CSRT-0.872371



Hình 17: KCF-0.542766

2. FaceOcc1 (IoU cao nhất của thuật toán KCF)

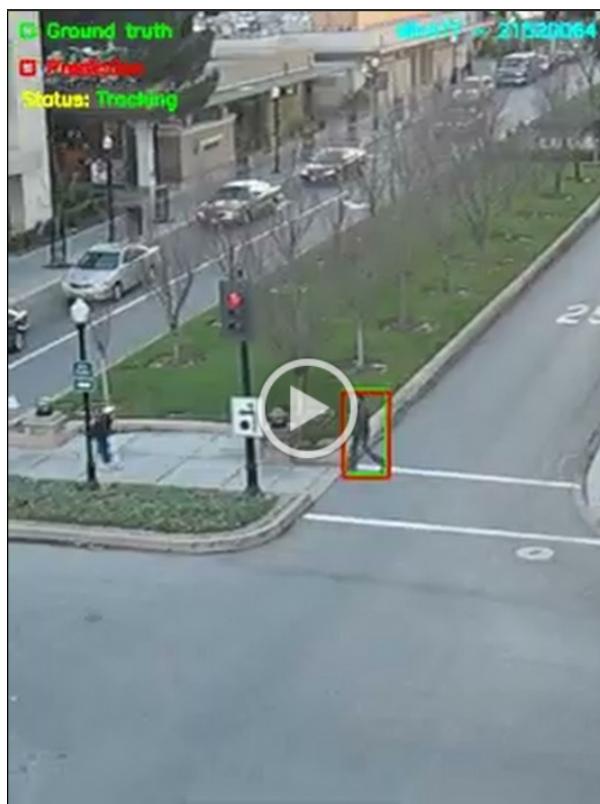


Hình 18: CSRT-0.574012

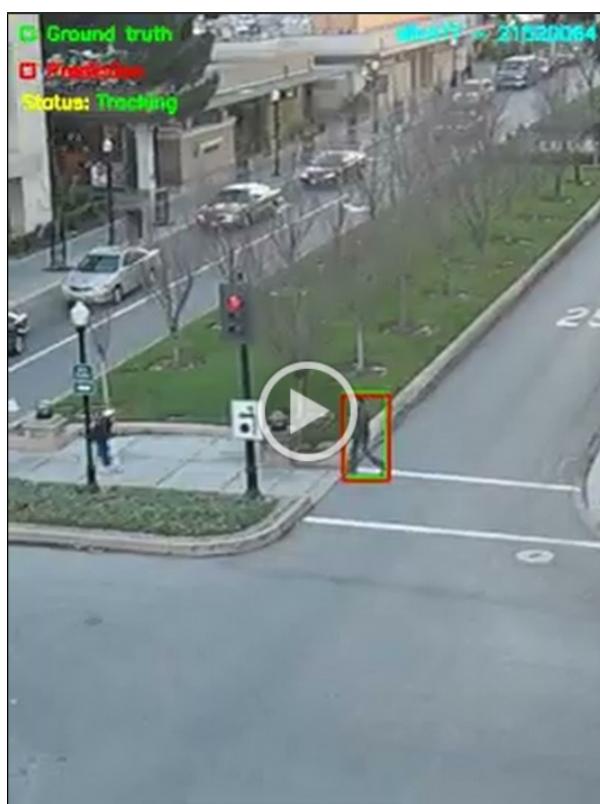


Hình 19: KCF-0.760306

3. Human3 (IoU thấp nhất của cả thuật toán KCF và CSRT)



Hình 20: CSRT-0.015626



Hình 21: KCF-0.002577

6.2 Demo trên các video tự cung cấp

Sau khi thử nghiệm một số chuỗi ảnh trên bộ dataset OTB2015, bây giờ mình sẽ thử trên một video bất kì mà mình upload lên từ máy.

Ví dụ, ở đây mình chạy video **mot.mp4**



Hình 22: Video mình đưa vào

Sau khi cho video chạy trên 2 thuật toán KCF và CSRT, ta thu được kết quả như bên dưới:



Hình 23: CSRT



Hình 24: KCF

7 Các link liên quan tới báo cáo

Các tài liệu liên quan tới đồ án nằm trong link github:

<https://github.com/trthminh/CS231.N21.KHTN>