Sample Code

Tracking Multiple Objects or Rectangles in Video

Apply Vision algorithms to track objects or rectangles throughout a video.

Download

iOS 12.0+ | iPadOS 12.0+ | Xcode 11.3+

Overview

With the Vision framework, you can detect and track objects or rectangles through a sequence of frames coming from video, live capture, or other sources.

This sample app shows you how to pick an initial object to track, how to create Vision tracking requests to follow that object, and how to parse results from the object or rectangle tracker.

Preview the Sample App

To see this sample app in action, build and run the project in Xcode, then choose a video from your photo library. Once the video is loaded from your photo library, choose to track either objects or rectangles.

Nominate Objects or Rectangles to Track

To track rectangles, select Rectangles. The app runs the rectangle detector and shows rectangles it finds in a preview of the scene.

Otherwise, to track objects, select Objects. Then nominate objects to track by touching them in the preview and dragging boxes around them. You can select multiple objects; the app identifies them by their `UUID`, using differently colored rectangles.

The VNTrackObjectRequest class requires a detected object observation to initialize. The sample provides this observation by running VNDetectRectanglesRequest, or by creating one from the bounding box you drew in the preview. It tracks multiple objects by iterating through each observation and creating a VNDetectedObjectObservation from its bounding box.

```
var inputObservations = [UUID: VNDetectedObjectObservation]()
var trackedObjects = [UUID: TrackedPolyRect]()
switch type {
case .object:
    for rect in self.objectsToTrack {
        let inputObservation = VNDetectedObjectObservation(boundingBox: rect.boundir
        inputObservations[inputObservation.uuid] = inputObservation
        trackedObjects[inputObservation.uuid] = rect
    }
case .rectangle:
    for rectangleObservation in initialRectObservations {
        inputObservations[rectangleObservation.uuid] = rectangleObservation
        let rectColor = TrackedObjectsPalette.color(atIndex: trackedObjects.count)
        trackedObjects[rectangleObservation.uuid] = TrackedPolyRect(observation: rec
    }
}
```

In your own app, if you prefer to nominate objects programmatically, you can use observations returned from Vision's own object detection algorithms. For example, the Vision framework's VNImageRequestHandler accepts face detection, text detection, and barcode detection requests, and those requests return their results in subclasses of VNObservation. You can pass these observations directly into VNTrackObjectRequest.

Selecting a salient object heavily influences the performance of the tracking algorithm; provide the best initial bounding box segmentation of your object that you can.

Track Objects or Rectangles with a Request Handler

The Vision framework handles tracking requests through a VNSequenceRequestHandler. Whereas the VNImageRequestHandler handles object detection requests on a still image, the VNSequenceRequestHandler handles tracking requests.

Create a tracking request for each rectangle or object you'd like to track. Seed each tracking request with the observation created during nomination.

```
let request: VNTrackingRequest!
switch type {
```

```
case .object:
    request = VNTrackObjectRequest(detectedObjectObservation: inputObservation.value
case .rectangle:
    guard let rectObservation = inputObservation.value as? VNRectangleObservation el
        continue
    }
    request = VNTrackRectangleRequest(rectangleObservation: rectObservation)
}
request.trackingLevel = trackingLevel

trackingRequests.append(request)
```

For each such request, call the sequence request handler's `perform(_:)` method, making sure to pass in the video reader's orientation to ensure upright tracking. This method runs synchronously; use a background queue, such as `workQueue` in the sample code, so that the main queue isn't blocked while your requests execute.

```
try requestHandler.perform(trackingRequests, on: frame, orientation: videoReader.ori
```

By iterating through each selected object or rectangle, creating a tracking request from it, and calling `perform` on the request handler, Vision follows the object or rectangle over the image sequence and returns results through its `results` property.

Interpret Tracking Results

Access tracking results through the request's `results` property or its completion handler. A single tracking request represents a single tracked object in a one-to-one relationship. If a tracking request succeeds, its `results` property contains `VNDetectedObjectObservation` objects describing the tracked object's new location in the frame.

```
guard let results = processedRequest.results else {
    continue
}
guard let observation = results.first as? VNDetectedObjectObservation else {
    continue
}
// Assume threshold = 0.5f
let rectStyle: TrackedPolyRectStyle = observation.confidence > 0.5 ? .solid : .dashe
let knownRect = trackedObjects[observation.uuid]!
switch type {
case .object:
    rects.append(TrackedPolyRect(observation: observation, color: knownRect.color, s
case .rectangle:
    guard let rectObservation = observation as? VNRectangleObservation else {
        break
    }
    rects.append(TrackedPolyRect(observation: rectObservation, color: knownRect.colo
}
```

Use the observation's boundingBox to determine its location, so you can update your app or UI with the tracked object's new location. Also use it to seed the next round of tracking.

```
inputObservations[observation.uuid] = observation
```

In practice, you should periodically run a new tracking request with an updated set of input observations to capture objects that weren't present in your initial nomination frame. For instance, you could create a new `VNTrackObjectRequest` every ten frames.

See Also

Object tracking

{} Tracking the User's Face in Real Time

Detect and track faces from the selfie cam feed in real time.

`class VNTrackingRequest`

The abstract superclass for image-analysis requests that track unique features across multiple images or video frames.

`class VNTrackRectangleRequest`

An image-analysis request that tracks movement of a previously identified rectangular object across multiple images or video frames.

`class VNTrackObjectRequest`

An image-analysis request that tracks the movement of a previously identified object across multiple images or video frames.

`class VNDetectedObjectObservation`

An observation that provides the position and extent of an image feature that an image-analysis request detects.