Slovenská technická univerzita
Fakulta informatiky a informačných technológií
Ilkovičova 3, 842 19 Bratislava 4

**Bc. Lukáš Sekerák**

# Detection of objects in soccer

Study program: Information systems
Grade: 5
Subject: Computer vision
Supervisor: Ing. Wanda Benešová, PhD.
Date: 2015

# 1 DESCRIPTION OF APPLICATION

Application is using OOP concept, so functionality is divided to multiple classes. MVC pattern was used to solve most of dependencies. The main part of project is in the Soccer class. This class represent main controller and it is managing devices, detectors, windows, drawers. Next we will describe responsibility of every class.

## 1.1 MAIN CLASSES

A Drawer class is managing view of data. Data entities comes to this class and they are already classified. So base on their type, they are visible, drawed to the windows or they not. We are specifying draw color for every object and a way how to draw it. User can draw a boundary of object, or a type in text format, or a covered distance in text format too. Also this class is managing ROI area and decide what will be draw in this area. For the ROI area we can show histogram of colors, or compute pixels clusters. This last functionality is disabled (in comment) yet, it had debug purpose.

An ObjectDetector class is important class, it is classifying objects. We start with a mask, where we find contours. Every contour is classified based on its position, size of area, count of pixels and by color. This classification is using some constants, constants are defined in header file of this class.

In an ObjectTracer class is section for tracing objects. Based on the track record it is building (in frames) history of traced object. At the start, algorithm is pre-processing the input image, also it is remembering previous frame input. Then we are comparing previous detected objects (its points), with a detected objects in current frame (mapped points). We are comparing them by intersection, we count pixels in same area. It is not best method, but suitable and give us good result. We are mapping old pixels to the new by calcOpticalFlowPyrLK algorithm.

- We discovered that only 0.1% points are lost with this sparse feature algorithm.
- When we find same object in next frame, the already used pixels (for mapping) are replaced with new pixels. This method guarantee us very good precision of tracking. However, this replace operation could be run every 10 frame.
- There is a filter method, where we define which objects we can track.
- Another method define which objects we can start track. Tracking of object can start manually, by a click on object, or automatically every new discovered object. We test and prefer manual way.

A Soccer class is most important. It is managing flow of the data, stream, learning process and image processing process. Every frame is of course preprocessed, for example we are resizing input. Then we are using learned BackgroundSubtractorMOG2 algorithm to detect possible objects (we detect mask only). We are segmenting playground too (by a color). Then we use erode, dilate operations and combine this masks to get best final mask. Final mask is created for detector of objects. This class extend App class, so it can read input, and bind pushing of keys. We hook pushing of keys to control application, for example we can pause video (stream) processing.

An Entities file is holding all data objects. Data objects are used across a project. Like a Frame or a FrameObject. The list of FrameObjects is result of a detector of objects. FrameObject has rectangle area, contour and type. We support several kind of types. Detected object, FrameObject can have a history, so there can be reference to the detected object in previous frame.

## 1.2 UTILITY CLASSES

A VideoRecord class is responsible for loading a video file. The video is sequence of frames, we read frame after frame in a stream. It is possible to restart this stream. We used this restart command after training algorithm. In this section we should read annotation file and map information for every frame, we are not doing it yet.

A Time class is small class to get exact and thread-safe time, it is used in locking of FPS.

A ThresholdColor class represent interval of color. This help class can tell us, how many pixels are in the mask, in this range. There is great method to register track bars for any window. This is useful to find possible interval.

A Main.cpp file is holding log4j configuration and an initialization of application.

An App class is responsible for communication with an Operating system. There is main cycle of application, calculations to limit fps, methods to stop application.

An Util file hold lot of utility stuff. Like a computing of pixel clusters, a computing of histograms. There are also mathematical methods to check intersection and methods to compute area size and a distance. Some of this stuff were copied from Open CV examples.

# 2 WORK FLOW

1. Read next frame
2. Pre-process next frame (resize input)
3. Learning section
   3.1. Learn Background sub-tractor algorithm
4. Detection section
   4.1. Compute masks
      4.1.1. Compute front mask from Background sub-tractor algorithm
      4.1.2. Segmentation of playground to mask
      4.1.3. Combine masks, erode, dilate
   4.2. Detect objects
      4.2.1. Find contours
      4.2.2. Determine objects
      4.2.3. Determine persons
         4.2.3.1. Determine teams
   4.3. Trace objects
      4.3.1. Trace saved objects
         4.3.1.1. Filter objects
         4.3.1.2. Map old pixels to new pixels
         4.3.1.3. Compare new pixels to new detected objects
      4.3.2. Check for new objects
   4.4. Draw data
      4.4.1. Determine visibility, color
      4.4.2. Choose draw method
      4.4.3. Draw ROI area
         4.4.3.1. Compute clusters
         4.4.3.2. Compute color histogram