CS5543 Real -Time Big Data Analytics
Project Report 1
Read Video

Akhilesh Gattu (5) Latha Muddu (15)

Project Objectives

Significance

To maintain a track record of each player and describe the swimming style. As many people are not aware of various swimming styles and the basic rules of swimming its being difficult to understand the swimming videos. So we would like to come up with an approach describing the swimming style of each player and maintaining track record of each player.

Features

1) Identifying the key frames
2) Creating highlighted video
3) Maintaining track record of the player
4) Recognizing swimming style

Approach

Data Sources: Youtube, TRECVID

Analytic Tools: Spark and Storm

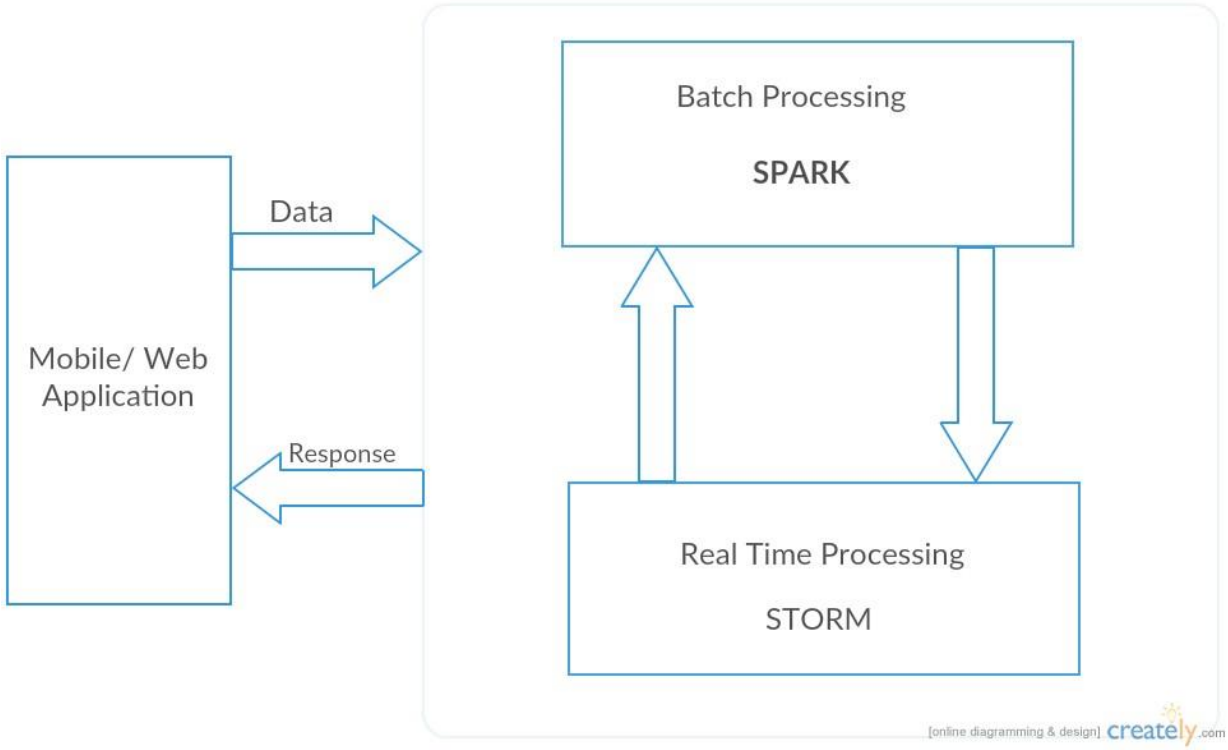
Analytical Tasks: Processing each and every frame individually, tracking each player in Storm Bolt.

Expected Input: Swimming Video

Expected Output: Mainframes, swimming style, highlighted video, track record of each player.

Application Specification

Software Architecture:



Technologies:

1) Android (Mobile Application)
2) Spark (Batch Processing, Model)
3) Storm (Real Time Streaming)
4) Kafka (Communication)

Storm:

Storm is a distributed real time system that can process large amounts of data with high velocity. The transmission rate of data in Storm is about million records per second. Storm uses Spouts and bolts to transmit data. The storm topology is like a directed acyclic graph with spouts and bolts as the vertices for the graph.

The characteristics of storm includes:

Fast: Processing of data is very fast and simple.

Scalable: In storm data is scalable as it supports parallel computations which can run across cluster of machines.

Fault-Tolerance: The system is fault tolerant because when the worker node dies, Storm will automatically restart them i.e., if the worker node dies, on the other node the worker will automatically be restarted.

Reliable: In storm the tuples will be processed at least once or exactly once. The messages will not be processed until unless the tuple fails in processing.

Easy to operate: Deployment of data is easy in storm.

Kafka:

Kafka is an open source stream data processing platform developed by apache which is written in Scala and java. Kafka is a distributed message broker system that can handle large amounts of data per second.

The streaming of data has the following capabilities:

1. We can publish and subscribe to streams of data or records.
2. The streams of records can be stored in a fault tolerant way.
3. The streaming is done as soon as the record occurs.

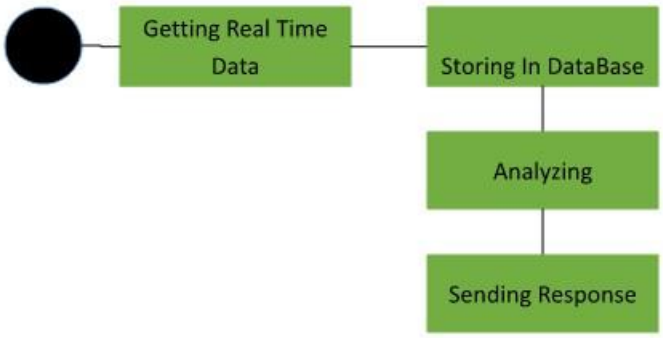
Kafka is best for two kinds of applications:

1. To build real time stream data pipelines which can fetch the data reliably between the systems.
2. To build real time streaming applications which can transform the streams of data easily.

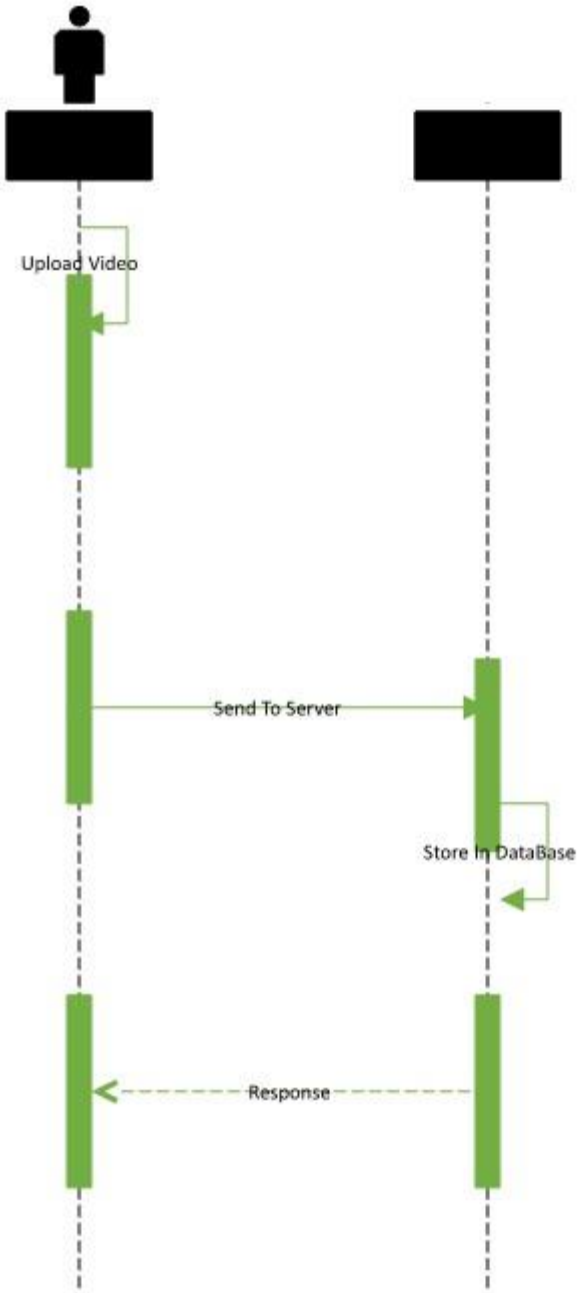
Kafka uses 4 API's mainly

1. Producer API – used to publish streams of records to the kafka topics
2. Consumer API – Allows an application to subscribe to one or more topic and process them.
3. Streams API – This API allows to consume an input stream for one or more topics and produce an output stream to one or more topics accordingly.
4. Connect API – Allows to build and reuse the producers or consumers that connects the kafka topics.

Activity Diagram:



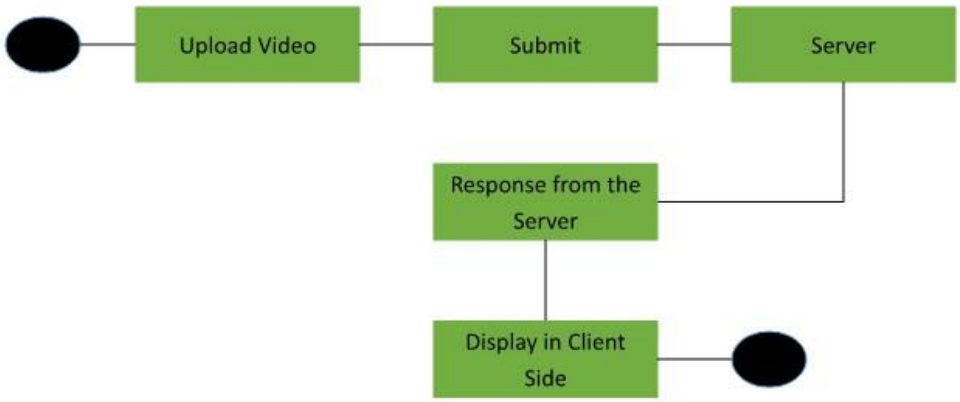
Sequence Diagram:



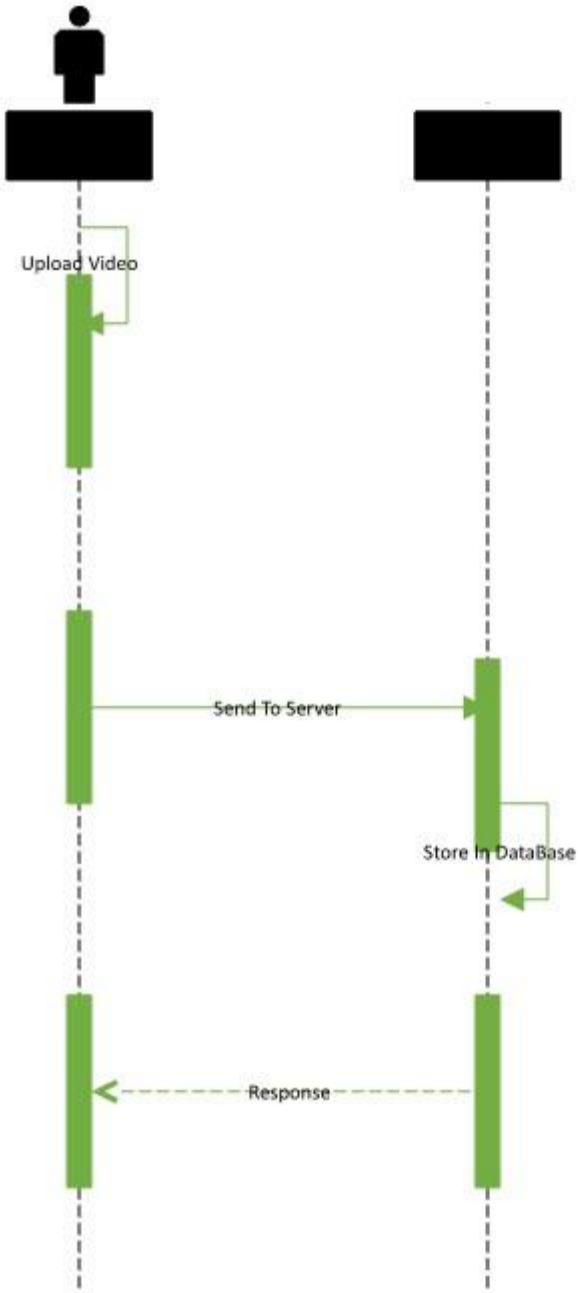
Operation Specification

Input: Video
Output: Text Description, Video

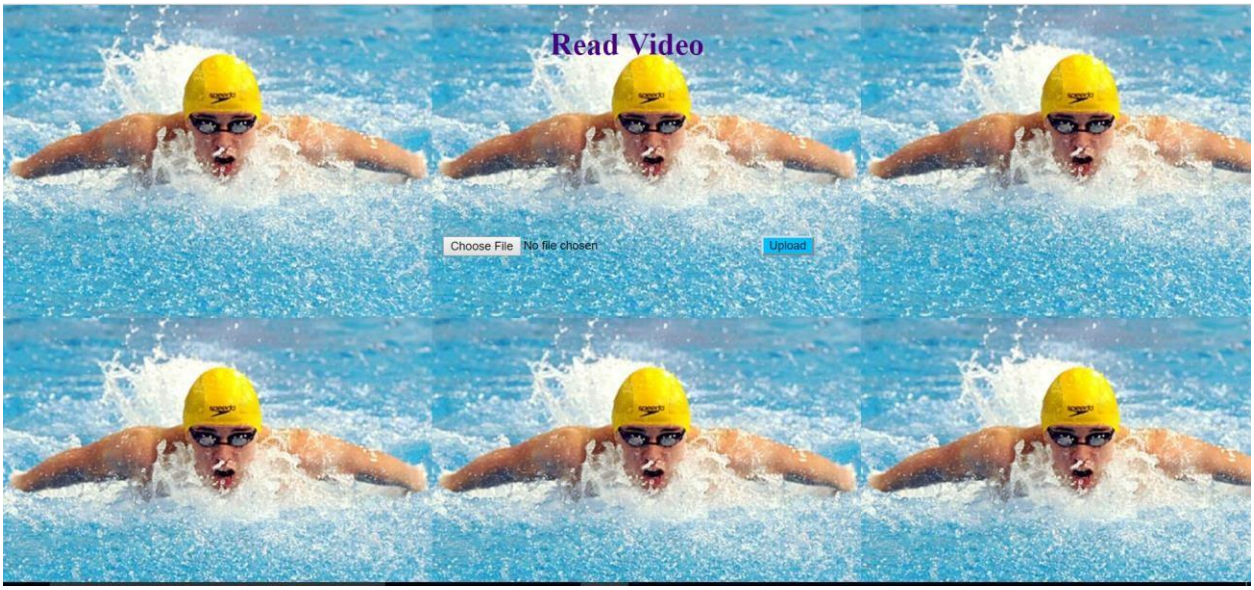
Activity Diagram:

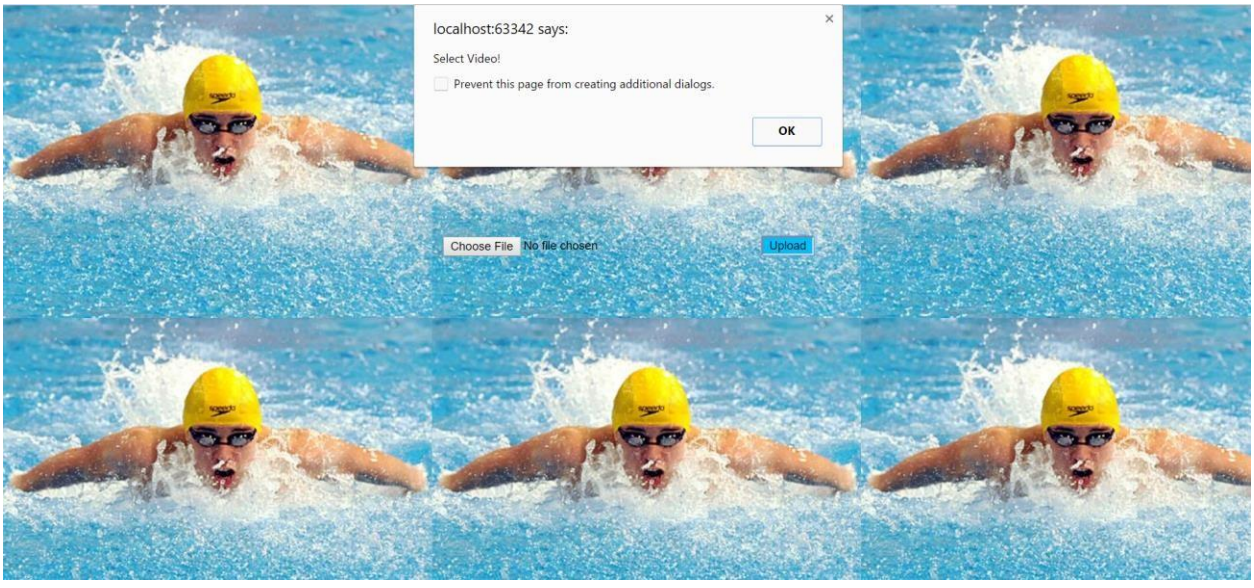


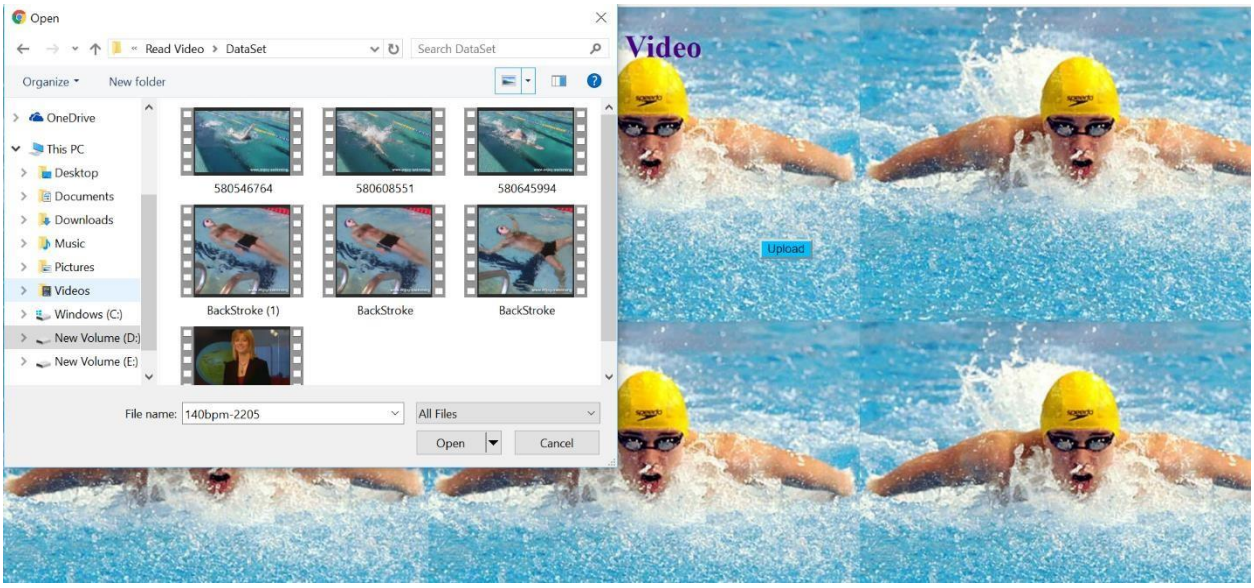
Sequence Diagram:

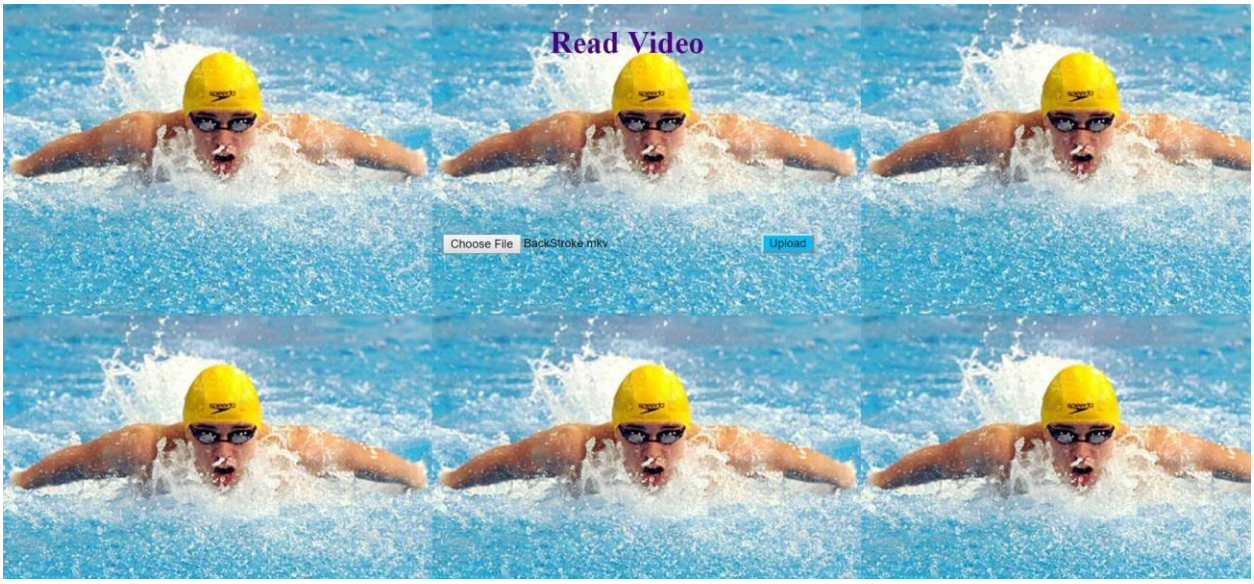


Documentation






Open
Read Video › DataSet
Search DataSet
Organize
New folder
OneDrive
This PC
Desktop
Documents
Downloads
Music
Pictures
Videos
Windows (C:)
New Volume (D:)
New Volume (E:)
580546764
580608551
580645994
BackStroke (1)
BackStroke
BackStroke
File name: 140bpm-2205
All Files
Open
Cancel
Video
Upload


Read Video
Choose File
BackStroke.mkv
Upload

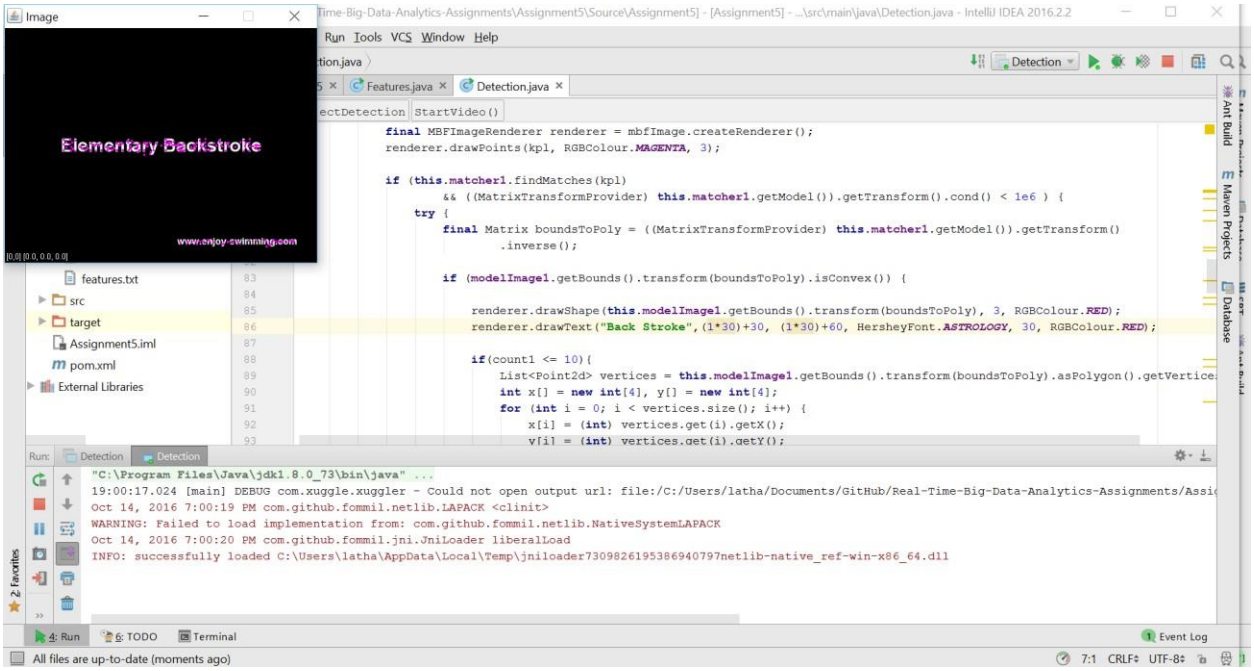


Image
Elementary Backstroke
www.enjoy-swimming.com
Time-Big-Data-Analytics-Assignments\Assignment5\Source\Assignment5] - [Assignment5] - ...\src\main\java\Detection.java - IntelliJ IDEA 2016.2.2
Run Tools VCS Window Help
Detection
Features.java
Detection.java
final MBFImageRenderer renderer = mbfImage.createRenderer();
renderer.drawPoints(kpl, RGBColour.MAGENTA, 3);
if (this.matcher1.findMatches(kpl)
&& ((MatrixTransformProvider) this.matcher1.getModel()).getTransform().cond() < 1e6) {
try {
final Matrix boundsToPoly = ((MatrixTransformProvider) this.matcher1.getModel()).getTransform()
.inverse();
if (modelImage1.getBounds().transform(boundsToPoly).isConvex()) {
renderer.drawShape(this.modelImage1.getBounds().transform(boundsToPoly), 3, RGBColour.RED);
renderer.drawText("Back Stroke",(1*30)+30, (1*30)+60, HersheyFont.ASTROLOGY, 30, RGBColour.RED);
if(count1 <= 10){
List<Point2d> vertices = this.modelImage1.getBounds().transform(boundsToPoly).asPolygon().getVertice
int x[] = new int[4], y[] = new int[4];
for (int i = 0; i < vertices.size(); i++) {
x[i] = (int) vertices.get(i).getX();
features.txt
src
target
Assignment5.iml
pom.xml
External Libraries
Run: Detection
"C:\Program Files\Java\jdk1.8.0_73\bin\java" ...
19:00:17.024 [main] DEBUG com.xuggle.xuggler - Could not open output url: file:/C:/Users/latha/Documents/GitHub/Real-Time-Big-Data-Analytics-Assignments/Assi
Oct 14, 2016 7:00:19 PM com.github.fommil.netlib.LAPACK <clinit>
WARNING: Failed to load implementation from: com.github.fommil.netlib.NativeSystemLAPACK
Oct 14, 2016 7:00:20 PM com.github.fommil.jni.JniLoader liberalLoad
INFO: successfully loaded C:\Users\latha\AppData\Local\Temp\jniloader7309826195386940797netlib-native_ref-win-x86_64.dll
4: Run
6: TODO
Terminal
Event Log
All files are up-to-date (moments ago)
7:1 CRLF UTF-8

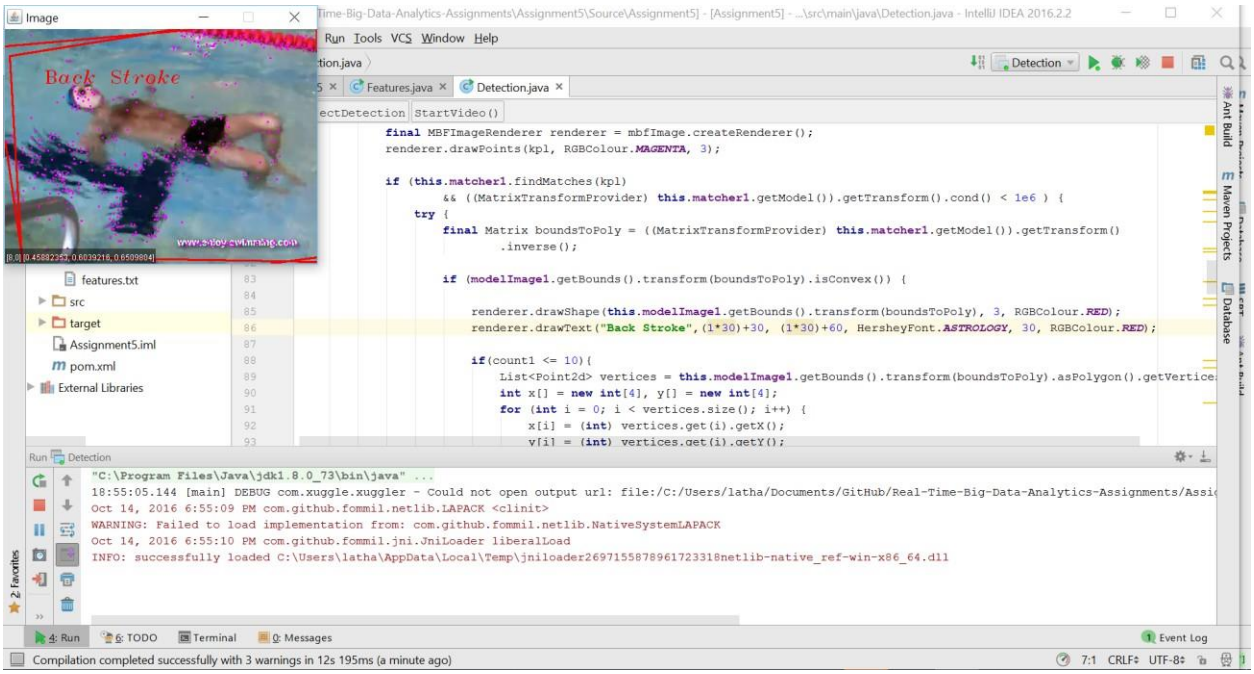


Image
Back Stroke
www.enjoy-swimming.com
Time-Big-Data-Analytics-Assignments\Assignment5\Source\Assignment5] - [Assignment5] - ...\src\main\java\Detection.java - IntelliJ IDEA 2016.2.2
Run Tools VCS Window Help
Detection
Features.java
Detection.java
final MBFImageRenderer renderer = mbfImage.createRenderer();
renderer.drawPoints(kpl, RGBColour.MAGENTA, 3);
if (this.matcher1.findMatches(kpl)
&& ((MatrixTransformProvider) this.matcher1.getModel()).getTransform().cond() < 1e6) {
try {
final Matrix boundsToPoly = ((MatrixTransformProvider) this.matcher1.getModel()).getTransform()
.inverse();
if (modelImage1.getBounds().transform(boundsToPoly).isConvex()) {
renderer.drawShape(this.modelImage1.getBounds().transform(boundsToPoly), 3, RGBColour.RED);
renderer.drawText("Back Stroke",(1*30)+30, (1*30)+60, HersheyFont.ASTROLOGY, 30, RGBColour.RED);
if(count1 <= 10){
List<Point2d> vertices = this.modelImage1.getBounds().transform(boundsToPoly).asPolygon().getVertice
int x[] = new int[4], y[] = new int[4];
for (int i = 0; i < vertices.size(); i++) {
x[i] = (int) vertices.get(i).getX();
features.txt
src
target
Assignment5.iml
pom.xml
External Libraries
Run Detection
"C:\Program Files\Java\jdk1.8.0_73\bin\java" ...
18:55:05.144 [main] DEBUG com.xuggle.xuggler - Could not open output url: file:/C:/Users/latha/Documents/GitHub/Real-Time-Big-Data-Analytics-Assignments/Assi
Oct 14, 2016 6:55:09 PM com.github.fommil.netlib.LAPACK <clinit>
WARNING: Failed to load implementation from: com.github.fommil.netlib.NativeSystemLAPACK
Oct 14, 2016 6:55:10 PM com.github.fommil.jni.JniLoader liberalLoad
INFO: successfully loaded C:\Users\latha\AppData\Local\Temp\jniloader2697155878961723318netlib-native_ref-win-x86_64.dll
4: Run
6: TODO
Terminal
0: Messages
Event Log
Compilation completed successfully with 3 warnings in 12s 195ms (a minute ago)
7:1 CRLF UTF-8

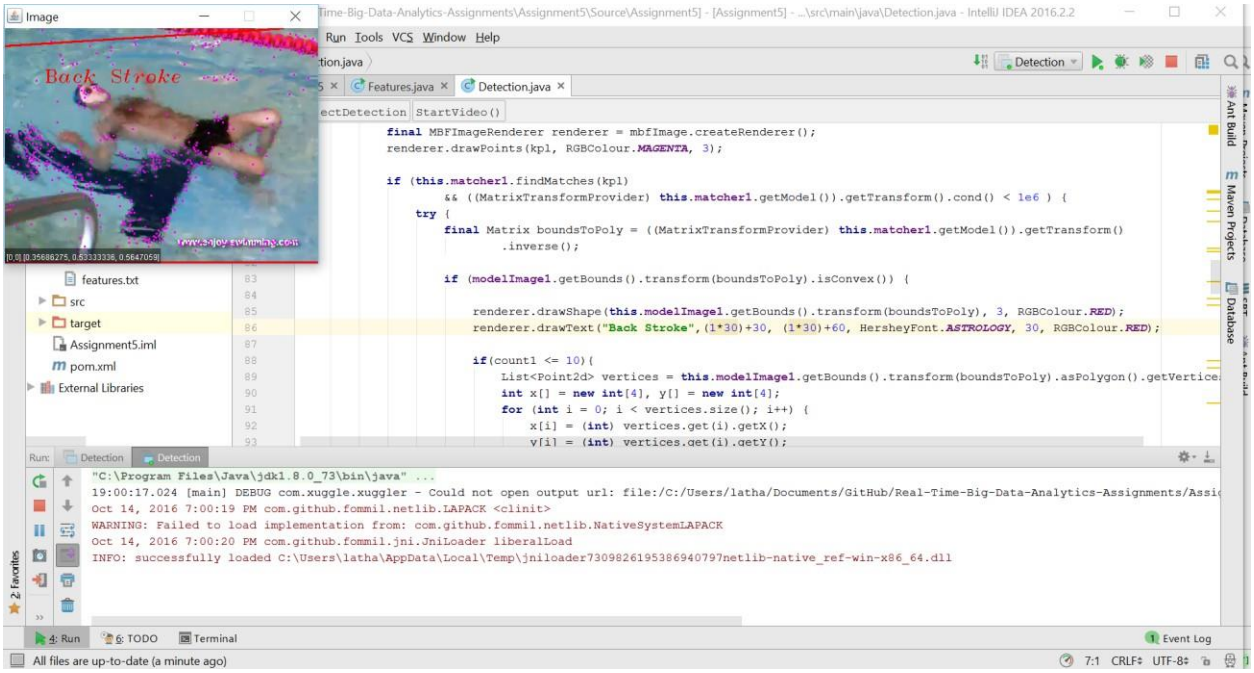


Image
Back Stroke
Run Tools VCS Window Help
Features.java
Detection.java
final MBFImageRenderer renderer = mbfImage.createRenderer();
renderer.drawPoints(kpl, RGBColour.MAGENTA, 3);
if (this.matcher1.findMatches(kpl)
&& ((MatrixTransformProvider) this.matcher1.getModel()).getTransform().cond() < 1e6) {
try {
final Matrix boundsToPoly = ((MatrixTransformProvider) this.matcher1.getModel()).getTransform()
.inverse();
if (modelImage1.getBounds().transform(boundsToPoly).isConvex()) {
renderer.drawShape(this.modelImage1.getBounds().transform(boundsToPoly), 3, RGBColour.RED);
renderer.drawText("Back Stroke",(1*30)+30, (1*30)+60, HersheyFont.ASTROLOGY, 30, RGBColour.RED);
if(count1 <= 10){
List<Point2d> vertices = this.modelImage1.getBounds().transform(boundsToPoly).asPolygon().getVertice
int x[] = new int[4], y[] = new int[4];
for (int i = 0; i < vertices.size(); i++) {
x[i] = (int) vertices.get(i).getX();
features.txt
src
target
Assignment5.iml
pom.xml
External Libraries
Run: Detection
"C:\Program Files\Java\jdk1.8.0_73\bin\java" ...
19:00:17.024 [main] DEBUG com.xuggle.xuggler - Could not open output url: file:/C:/Users/latha/Documents/GitHub/Real-Time-Big-Data-Analytics-Assignments/Assi
Oct 14, 2016 7:00:19 PM com.github.fommil.netlib.LAPACK <clinit>
WARNING: Failed to load implementation from: com.github.fommil.netlib.NativeSystemLAPACK
Oct 14, 2016 7:00:20 PM com.github.fommil.jni.JniLoader liberalLoad
INFO: successfully loaded C:\Users\latha\AppData\Local\Temp\jniloader7309826195386940797netlib-native_ref-win-x86_64.dll
4: Run
6: TODO
Terminal
Event Log
All files are up-to-date (a minute ago)
7:1 CRLF UTF-8

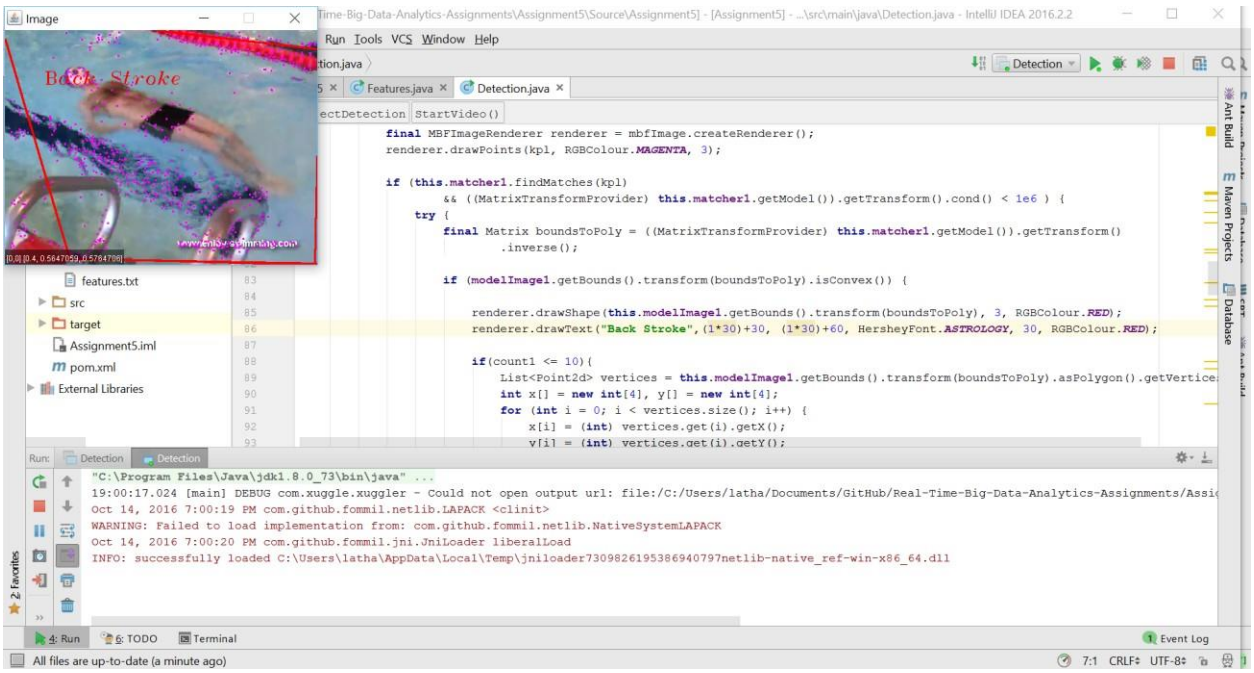


Image
Back Stroke
Run Tools VCS Window Help
Features.java
Detection.java
final MBFImageRenderer renderer = mbfImage.createRenderer();
renderer.drawPoints(kpl, RGBColour.MAGENTA, 3);
if (this.matcher1.findMatches(kpl)
&& ((MatrixTransformProvider) this.matcher1.getModel()).getTransform().cond() < 1e6) {
try {
final Matrix boundsToPoly = ((MatrixTransformProvider) this.matcher1.getModel()).getTransform()
.inverse();
if (modelImage1.getBounds().transform(boundsToPoly).isConvex()) {
renderer.drawShape(this.modelImage1.getBounds().transform(boundsToPoly), 3, RGBColour.RED);
renderer.drawText("Back Stroke",(1*30)+30, (1*30)+60, HersheyFont.ASTROLOGY, 30, RGBColour.RED);
if(count1 <= 10){
List<Point2d> vertices = this.modelImage1.getBounds().transform(boundsToPoly).asPolygon().getVertice
int x[] = new int[4], y[] = new int[4];
for (int i = 0; i < vertices.size(); i++) {
x[i] = (int) vertices.get(i).getX();
features.txt
src
target
Assignment5.iml
pom.xml
External Libraries
Run: Detection
"C:\Program Files\Java\jdk1.8.0_73\bin\java" ...
19:00:17.024 [main] DEBUG com.xuggle.xuggler - Could not open output url: file:/C:/Users/latha/Documents/GitHub/Real-Time-Big-Data-Analytics-Assignments/Assi
Oct 14, 2016 7:00:19 PM com.github.fommil.netlib.LAPACK <clinit>
WARNING: Failed to load implementation from: com.github.fommil.netlib.NativeSystemLAPACK
Oct 14, 2016 7:00:20 PM com.github.fommil.jni.JniLoader liberalLoad
INFO: successfully loaded C:\Users\latha\AppData\Local\Temp\jniloader7309826195386940797netlib-native_ref-win-x86_64.dll
4: Run
6: TODO
Terminal
Event Log
All files are up-to-date (a minute ago)
7:1 CRLF UTF-8

Project Management

Plan:
Increment 1: Key Frame Detection and Object Tracking
Increment 2: Client side UI
Increment 3: Working with Storm and Spark
Increment 4: Communication between client and server

Work Completed: For the third increment we were able to send features to storm, build decision tree model in spark and save it in the mongo database. We were also able to extract the model in storm and able to build the storm topology dynamically.

Responsibility: Both of us have divided the tasks equally and worked on it. At the end of each task we shared our knowledge in that task.

Contributions: Both of us worked on all the tasks equally.

Akhilesh Gattu: 50

Latha Muddu: 50

Work to be Completed:
For the upcoming increments the primary task that needs to be completed is the server side development and at the end integrating both the client and server implementations.

Submission & Deployment

References:

https://kafka.apache.org/documentation.html#producerapi

http://hortonworks.com/apache/storm/

https://en.wikipedia.org/wiki/Apache_Kafka

Github Link:

https://github.com/AkhileshGattu/Read-Video/tree/master/Source