TRECVid 2008 Event Detection:
ViPER XML Representation of Events

Introduction

The TRECVid 2008 event detection evaluation will use XML to encode events for annotations/system output. In particular, the evaluation is using the ViPER (Video Performance Evaluation Resource) XML format.[1] ViPER has previously been used for spatio-temporal annotation of objects in video, and has online documentation available at: *http://viper-toolkit.sourceforge.net/docs/*

NIST has developed a corresponding XML Schema Definition (XSD) to provide a uniform syntactic check for all annotations and system output. The top-level XSD, **TrecVid08.xsd**, depends on two others, **TrecVid08-viper.xsd** and **TrecVid08-viperdata.xsd**. The top-level XSD is as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
        <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
        xmlns:viper="http://lamp.cfar.umd.edu/viper#"
        xmlns:viperdata="http://lamp.cfar.umd.edu/viperdata#">
        <xsd:import namespace="http://lamp.cfar.umd.edu/viper#" schemaLocation=
        "TrecVid08-viper.xsd"/>
        <xsd:import namespace="http://lamp.cfar.umd.edu/viperdata#" schemaLocation=
        "TrecVid08-viperdata.xsd"/>
</xsd:schema>
```

Together, the **TrecVid08-viper.xsd** and **TrecVid08-viperdata.xsd** define the allowable data types in a ViPER XML file that conforms to this top-level XSD. All XSD files (and sample XML files) are available for review and comment at the following Web site: *http://www.nist.gov/speech/tests/trecvid/2008/*

In the following sections we provide further details about the ViPER XML format, event representation, and system output files/annotations.

ViPER XML format details

The ViPER hierarchy is divided into “config” and “data” sections. Thus, all metadata files will contain a “config” and a “data” section. The config section consists of two descriptors: one to represent the relevant source file, and one descriptor to define the event. This allows properties of the source file and the event type to be defined at the top-level of the file (config), and organizes specific details about the source file and all event instances in the “data” section. The source file descriptor will include information about its file location, number of frames, frame rate, etc. The event descriptor indicates that events are represented as ViPER objects and it includes the following attributes: Point, BoundingBox, DetectionScore, and DetectionDecision. For the 2008 evaluation, only the

[1] D. Doermann and D. Mihalcik. Tools and Techniques for Video Performance Evaluation. In *ICPR*, volume 4, pages 167–170, 2000.

data for DetectionScore and DetectionDecision will be considered as input for scoring. However, each attribute (Point, BoundingBox, DetectionScore, and DetectionDecision) is required for all events, for both system output and annotation files. An example XML fragment illustrating the **config** section and an **event descriptor** is as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<viper xmlns="http://lamp.cfar.umd.edu/viper#" xmlns:data="http://lamp.cfar.umd.edu/viperdata#">
        <config>
                <descriptor name="Information" type="FILE">
                …
                </descriptor>
                <descriptor name="DoorOpenClose" type="OBJECT">
                <attribute dynamic="true" name="Point"
                type="http://lamp.cfar.umd.edu/viperdata#point"/>
                <attribute dynamic="true" name="BoundingBox"
                type="http://lamp.cfar.umd.edu/viperdata#bbox"/>
                <attribute dynamic="true" name="DetectionScore"
                type="http://lamp.cfar.umd.edu/viperdata#fvalue"/>
                 <attribute dynamic="true" name="DetectionDecision"
                type="http://lamp.cfar.umd.edu/viperdata#bvalue"/>
                </descriptor>
        </config>
        …
</viper>
```

An example XML fragment illustrating the data section for the relevant **source file** is as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<viper xmlns="http://lamp.cfar.umd.edu/viper#" xmlns:data="http://lamp.cfar.umd.edu/viperdata#">
        <config>
                 …
        </config>
         <data>
              <sourcefile filename="file:/usr/local/video/20050519-1503-Excerpt.mpg">
                <file id="0" name="Information">
                <attribute name="SOURCETYPE"/>
                 <attribute name="NUMFRAMES"> <data:dvalue value="5121"/>
                </attribute>
                <attribute name="FRAMERATE">
                 <data:fvalue value="1.0"/>
                </attribute>
                <attribute name="H-FRAME-SIZE"/>
                <attribute name="V-FRAME-SIZE"/>
                </file>
                …
                </sourcefile>
        </data>
</viper>
```

How events are represented

The config section of the metadata file specifes all event instances are ViPER objects. These instances are to appear in the data section, immediately following the source file. Event instances are subordinate to the source file data element, and importantly, each instance includes the event's *name*, a consecutively numbered *id*, and *framespan* indicating temporal extent. **Note that in ViPER, time begins at 0.0, while frame numbers begin with frame 1.**

Thus, an example XML fragment illustrating an **event** is as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<viper xmlns="http://lamp.cfar.umd.edu/viper#" xmlns:data="http://lamp.cfar.umd.edu/viperdata#">
        <config>
                …
        </config>
         <data>
                <sourcefile filename="file:/usr/local/video/20050519-1503-Excerpt.mpg">
                        …
                        <object framespan="1493:1557" id="0" name="DoorOpenClose">
                        <attribute name="Point"/>
                        <attribute name="BoundingBox"/>
                        <attribute name="DetectionScore"/>
                        <attribute name="DetectionDecision"/>
                        </object>
                        …
                </sourcefile>
        </data>
</viper>
```

Comparison of system output to annotation file format

The key difference between system output and reference annotation files is that systems outputs will include data for the beginning and end frame numbers, a detection decision, and detection score for each event observation, whereas reference annotations will include data only for the beginning and end frame numbers. An example XML fragment illustrating a **system's event observations** will appear similar to the following:

```
<?xml version="1.0" encoding="UTF-8"?>
<viper xmlns="http://lamp.cfar.umd.edu/viper#" xmlns:data="http://lamp.cfar.umd.edu/viperdata#">
        <config>
                …
        </config>
         <data>
                <sourcefile filename="file:/usr/local/video/20050519-1503-Excerpt.mpg">
                        …
                        <object framespan="1493:1557" id="0" name="DoorOpenClose">
                        <attribute name="Point"/>
                        <attribute name="BoundingBox"/>
                        <attribute name="DetectionScore"><data:fvalue value="0.887"/></attribute >
                        <attribute name="DetectionDecision"><data:bvalue value="true"/></attribute>
                        </object>
                        <object framespan="1776:1833" id="1" name="DoorOpenClose">
                        <attribute name="Point"/>
                        <attribute name="BoundingBox"/>
                        <attribute name="DetectionScore"><data:fvalue value="0.727"/></attribute >
                        <attribute name="DetectionDecision"><data:bvalue value="true"/><attribute/>
                        </object>
                        <object framespan="1950:2012" id="2" name="DoorOpenClose">
                        <attribute name="Point"/>
                        <attribute name="BoundingBox"/>
                        <attribute name="DetectionScore"><data:fvalue value="0.112"/></attribute >
                        <attribute name="DetectionDecision"><data:bvalue value="false"/></attribute>
                        </object>
                        …
                </sourcefile>
        </data>
</viper>
```

An example XML fragment illustrating the corresponding **annotation** (ground truth) file is:

```
<?xml version="1.0" encoding="UTF-8"?>
<viper xmlns="http://lamp.cfar.umd.edu/viper#" xmlns:data="http://lamp.cfar.umd.edu/viperdata#">
        <config>
                 …
        </config>
         <data>
                <sourcefile filename="file:/usr/local/video/20050519-1503-Excerpt.mpg">
                        …
                        <object framespan="1490:1560" id="0" name="DoorOpenClose">
                        <attribute name="Point"/>
                        <attribute name="BoundingBox"/>
                        <attribute name="DetectionScore"/>
                        <attribute name="DetectionDecision"/>
                        </object>
                        <object framespan="1770:1849" id="1" name="DoorOpenClose">
                        <attribute name="Point"/>
                        <attribute name="BoundingBox"/>
                        <attribute name="DetectionScore"/>
                        <attribute name="DetectionDecision"/>
                        </object>
                        …
                </sourcefile>
        </data>
</viper>
```

Systems may emit data for *Point* locations or *BoundingBox* information specific to each event instance. Both *Point* and *BoundingBox* data are illustrated in the following example:

```
        <object framespan="938:1493" id="0" name="DoorOpenClose">
                <attribute name="Point">
                        <data:point framespan="938:938" x="263" y="353"/>
                </attribute>
                <attribute name="BoundingBox">
                        <data:bbox framespan="1000:1100" height="101"  width="131" x="105"
                y="168"/>
                        <data:bbox framespan="1101:1400" height="89" width="166" x="99"
                y="168"/>
                </attribute>
                …
        </object>
```

Thus, each event observation may *optionally* have data for multiple point/bounding box locations, each of which has their own temporal extent, and are stored as child elements of the event observation.