

# Multimedia Annotation, Querying and Analysis in ANVIL

Michael Kipp  
DFKI, Saarbrücken, Germany  
kipp@dfki.de

## Abstract

While the availability of multimedia data, including human movement recording by motion capture, is steadily growing, the integrated viewing, annotation, and analysis of such complex data is still a challenge. The ANVIL tool, a widely used multi-track video and audio annotation tool, has now been extended to allow the synchronized handling of multiple media, especially the 3D viewing of motion capture data, to perform SQL queries and to conduct automated statistical analysis. The underlying database in conjunction with association detection allow analysis across tracks and modalities. This can be exploited in many contexts, from qualitative behavior analysis to the collection of training data for information extraction. Apart from describing the underlying methods and their realization in ANVIL, we discuss open issues like tool interoperability and scheme standardization.

## 1 Introduction

The goal of finding meaning in data has two extreme manifestations. In the empirical sciences, researchers attempt to interpret surface behaviors of humans or animals according to precise guidelines. In computer science, researchers search for automatic methods to extract meaning from low-level data. While the methods are different, they share the underlying data (video and audio files, motion capture data etc.) and the general aim to extract meaning. Both approaches can benefit from each other and are in fact often combined. The computer scientist needs to explore his/her data in a qualitative fashion to determine promising predictors and to build training and test corpora. The empirical scientist can use automatic, quantitative methods to bootstrap the manual annotation process and to increase the objectivity of the approach. Both kinds of research needs appropriate tools to support this process. For the qualitative annotation, browsing and analysis of videos, several tools have been developed [RLD<sup>+</sup>06, BLH01]. Each tool is usually derived from a rather specific research area but generalizable to a certain extent. In this paper, we present extensions to the ANVIL<sup>1</sup> video research tool [Kip08, MK02, Kip01] that makes a first step toward an integrated multimedia annotation, browsing and analysis platform. The extensions comprise of motion capture integration, database integration

---

<sup>1</sup><http://www.anvil-software.de>

and a number of analysis features. The driving force behind these extensions was the need to go beyond single modality, single media analysis and open up the capabilities to cut across modalities and media.

A common target of analysis is human behavior. For something as complex as human behavior, research has moved from performing unimodal analysis to multimodal analysis. Likewise, with an increased availability of capture and storage devices, researchers are moving from few media sources (e.g. a single video) to multiple media sources. For human motion, the most precise media is a *motion capture file* which can be acquired using various techniques (from complex marker-based optical systems to inexpensive inertia-based systems [RLS08]). Therefore, a complex data situation for human behavior analysis would consist of multiple video, audio and motion capture files [FvdKRvW08]. However, existing annotation tools cannot display motion capture data in an appropriate fashion, i.e. as an animated 3D skeleton. This is surprising since automatically processing motion capture data like e.g. cutting it into meaningful segments and classify different motion types (e.g. walking, standing, sitting etc.) is a common problem in computer graphics [BSP<sup>+</sup>04]. Motion capture data is very high-dimensional and contains no semantics, i.e. it is difficult to automatically interpret it. Therefore, a viewing and annotation tool is of use to start tackling this kind of data.

Apart from viewing the media, browsing the annotations can become tedious as the corpus is growing. Therefore, we integrated an SQL compliant database into ANVIL to have access to the powerful SQL query framework. This forms the backbone of association analysis where we want to look at categories on two different tracks and quantitatively evaluate whether they are *associated* and how. Other analysis features look at a single track and the *transition* of categories, visualized by a transition diagram which is in essence a Markov model. Ultimately, when moving to a full-fledged statistics tool, ANVIL offers various export functions, most notably framenumbers-by-attribute matrix that allows co-occurrence to be analyzed by external tools.

This paper is organized as follows. First, we briefly introduce the main concepts of the ANVIL tool and then discuss how a database and multiple media (especially 3D viewing of motion capture files) are integrated. Then, we look at two analysis features, namely transition diagrams and association analysis. We also talk about the export format for external statistical analysis. We conclude with a short survey of similar or related tools and close with a conclusion, discussing the burning issues of tool interoperability and scheme standardization.

## 2 Multi-Level Annotation

ANVIL is a tool for the systematic manual annotation of digital video [Kip01]. The user can transcribe events that occur in the video and put these transcriptions on parallel tracks running along the time axis (Figure 1). The transcription/description of a single event is called annotation *element* which is displayed as a box in one of the tracks, in time alignment with all other elements. In Figure 1, the annotation board is the bottom window, containing color-coded elements on the bottom-most tracks. The track layout can be fully specified by the user in a separate file, the so-called *coding scheme*, making the tool inde-

pendent of a specific research field or underlying theory (see [ACD<sup>+</sup>05][KNA07] for sample gesture coding schemes).

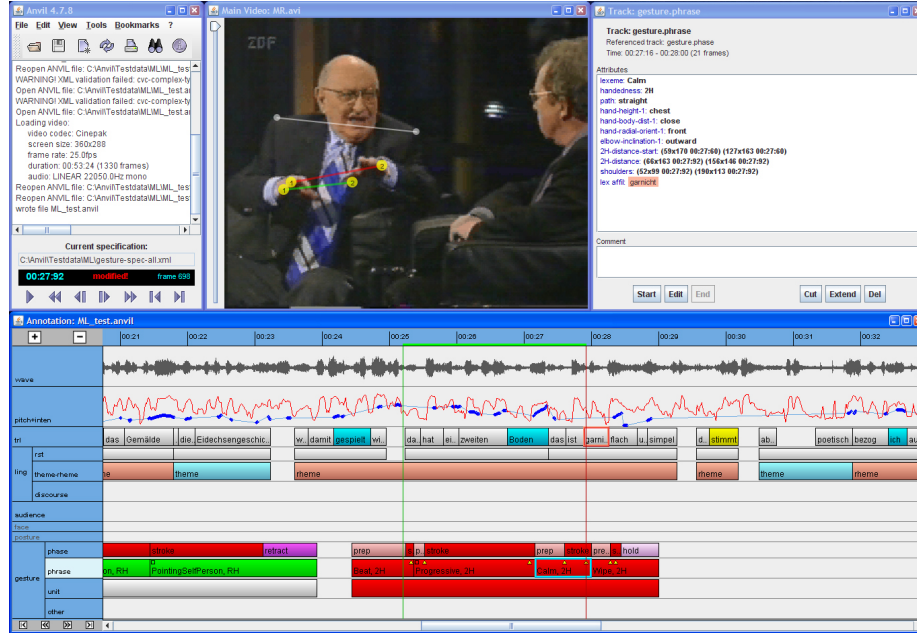


Figure 1: ANVIL graphical user interface. The bottom window, the so-called annotation board, is the main instrument of the coder who adds *elements* that appear as time-aligned boxes on screen.

In ANVIL, the annotation elements are actually *objects*, containing multiple information in *attributes*. This allows the encoding of a complex event like a gesture in a single box on screen while still having the event’s various aspects like gesture type, handedness, shape, etc. contained in this element. Furthermore, attributes are *typed* which means the user restricts the possible values of an attribute to, e.g., a certain set of labels (also called a *controlled vocabulary*) or to a range of numbers.

As for tracks, the underlying assumption is that all encodings in one track have similar properties. Therefore, for each track the set of corresponding attributes must be predefined by the user in the coding scheme. For example, for a track “gesture”, there could be two attributes “type” and “handedness”. Tracks come in various flavors to model the fact that a certain relationship holds between a track *A* and a reference track *B*. For instance, an element in *A* may always have a corresponding element in *B* with the exact same begin/end times. In this case, track *A* would be declared a *singleton type* track with reference track *B*. Another type is *span* where each element of track *A* consists of a sequence of elements in reference track *B*. The “spanning” element in track *A* inherits the begin time of the first element in this sequence and the end time of the last one. This inheritance of timestamps is the main advantage of track types: Both in the GUI and internally, the begin/end times of *singleton* and *span* type tracks are always inferred from the reference track, making manual alignment unnecessary and coding errors less likely.

Relationships between tracks reflect systematic relationships between their contained elements, in the above cases it is temporal correspondence or containment. However, one often needs to encode quite arbitrary relationships between encoded elements. ANVIL allows to do this in the form of *logical links*. A link is a special type of attribute that contains a list of links to other elements.

Elements in tracks have a start and end time as inherent properties. However, sometimes an element in a video exists for the whole duration of the video (e.g. an object on a table) or is not even concrete (a person). In ANVIL, one can encode such nontemporal entities in a data container called a *set* which is the equivalent of a track, just without time information [MK02]. A set is visualized using a simple table. In conjunction with logical links these elements allow the encoding of complex relations.

Although, ANVIL and similar tools are usually targeted at time-based information, for a number of applications it is not enough to encode *when* something happened, but it is also important *where* (on the video screen) it happened. Therefore, it is possible to perform *spatial coding* directly on the video screen [Kip08]. In Figure 1, the coder marked-up point locations on the video screen which are displayed as connected dots. The screen locations are encoded as timestamped screen coordinates in a special type of attribute.

Finally, since large corpus collection and annotation efforts usually consist of numerous media files and corresponding annotations, a *project tool* allows to group multiple annotation data files together if they are based on the same coding scheme. The project tool allows perform search and export operations over the whole corpus.

### 3 Database Integration

Multi-layer annotations of multiple media can quickly become cluttered, so that the user needs query tools to find information. Since ANVIL allows to package multiple bits of information into a single element, queries are even more important.

The SQL query language is not only very powerful but also an industry standard. Therefore, ANVIL internally maps the user's annotations to a temporary SQL database that is kept in sync at all times. Each track corresponds to a table: each annotation element is a row, each attribute a column (Figure 2). The user can now use the full expressive power of SQL to post queries. Since formulating such queries requires expert knowledge, we drafted a simplified syntax for the most basic queries: (a) finding elements in a single track using attribute constraints and (b) finding elements of two tracks that have a certain temporal relationship (e.g. overlap). For implementation we use the Java-based HSQL database engine<sup>2</sup>.

Two important restrictions we had to make is that our database integration does neither handle logical pointers nor explicitly model track relationships.

#### 3.1 Mapping Annotations to Database Tables

An annotation track has a lot in common with a table. A track represents a certain *type* of information that has various properties encoded in attributes.

---

<sup>2</sup><http://hsqldb.org>

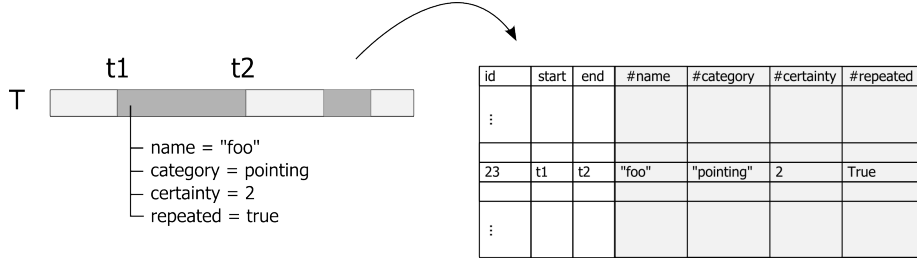


Figure 2: Each individual ANVIL track is mapped to its own DB table where every attribute is represented in a column and three special columns contain the primary key, start and end time.

A database table usually encodes properties in columns, while rows represent instances. Therefore, for each track, we create a specific table with columns for each attribute (Figure 2). Since tracks contain elements that have begin/end timestamps we have to store them in special columns. We avoid column name clashes by prefixing the user-defined attributes with a special symbol, a hash sign, to ensure that a user-defined attribute called “start” does not clash with our internal “start” property. The *id* column is our primary key to annotation elements. This ID is generated by ANVIL by increasing an internal counter each time an element is generated. Therefore, these IDs are unique across tracks. Query results can easily mapped back to ANVIL’s internal representation of the corresponding annotation elements. Note that the database tables must be kept in sync throughout the annotation session, i.e. deletions, additions and modifications on the annotation board must be reflected in the tables. When ANVIL shuts down the database is simply closed, to be recreated from scratch on next launch.

ANVIL type	SQL type
String	VARCHAR
Number	INTEGER
Float	FLOAT
Boolean	BOOLEAN
ValueSet	VARCHAR

Table 1: Mapping between ANVIL and SQL data types.

In the mapping depicted in Fig. 2, we have to convert ANVIL value types to SQL data types. For most types there was a corresponding type (e.g. SQL type *integer* for ANVIL type *number*), for all others we simply chose the SQL *varchar* type which is an arbitrary string of alphanumeric characters (Table 1).

### 3.2 Single-Track Queries

A query is a request for a subset of all annotation elements, given some constraints. The single-track query restricts this to a single track. Constraints can be formulated in SQL syntax depending on the SQL data type (Table 1): Strings can be queried using regular expressions, numbers can be queried with numeric

comparison operators (<, > etc.). Since SQL syntax must be learned and can quickly become tedious to write, we offer a simplified scripting language that allows to specify track plus a, possibly nested, combination of attribute constraints:

```
[ mytrack , ( att1 = 2H OR att1 = LH ) AND anotherAtt <> null ]
```

This is translated to the somewhat unwieldy SQL expression:

```
SELECT "mytrack"."id", "mytrack"."#att1", "mytrack"."#anotherAtt"
FROM "mytrack"
WHERE ("mytrack"."#att1" = '2H' OR "mytrack"."#att1" = 'LH')
AND "mytrack"."#anotherAtt" <> 'null'
```

The expression returns all elements in track `mytrack` which have value 2H or LH in `att1` and have a non-empty attribute called `anotherAtt`. In ANVIL, the returned IDs are used to collect the corresponding ANVIL elements.

### 3.3 Temporal Relationship Queries

One key interest of researchers using ANVIL lies in the relationship between elements of *different* tracks, comparing those which temporally coincide or have some other systematic temporal relationship. However, in order to analyze, for example, pair-wise relationships between elements of different tracks, one has to define under which conditions element  $E_1$  of track  $T_1$  and element  $E_2$  of track  $T_2$  should be compared. One way to do this is to let the user define the *temporal relation* that must hold so that two elements are comparable. We use the 7 Allen relations for this: equals, before, meets, overlaps, starts, finishes and during. In addition, we let the user specify a *tolerance* limit in seconds (a float value). For example, the relation (`equals`, `.4`) holds if the start time of element  $E_1$  and the start time of element  $E_2$  differ maximally `.4` seconds (and if the same holds for the end time).

Again, to spare the user from using long and complex SQL expressions we have a special syntax to ask for elements from two tracks that are characterized by a certain temporal relationship. An example is:

```
R[overlaps, .8] [ firstTrack, hand = 2H ] [ otherTrack, hand <> null ]
```

As one can see, this generalized the previously introduced example by using two single-track queries and defining a temporal relationship constraint on top.

Temporal relationship queries are the first step for analysis, either in the form of association analysis in ANVIL (Section 5.2) or in other types of analyses using external tools (see Section 5.3).

## 4 Integrating Motion Capture

ANVIL presupposes that a certain event was documented using multiple types and instances of media. For instance, psychotherapists interested in changes of facial expression and posture during a therapy session, may record such a session using multiple video cameras and microphones. Other media like biometric measurements, eye tracking and even motion capture may be added. In ANVIL,

the challenge is to allow for synchronized playback of multiple media streams. In particular, we wanted to integrate 3D motion capture playback which allows a fine-grained 3D reconstruction of human motion, and while it is nowadays most commonly used in computer animation, it has the potential of becoming the next generation tool for human behavior research.

## 4.1 Multiple Videos

Video playback in ANVIL is handled by the Java Media Framework (JMF), complemented by the JFFMPEG<sup>3</sup> package which adds a number of codecs. When playing multiple videos, the internal framework has to synchronize the different media using a single clock. Since in JMF each video is itself modeled as a clock, one video is declared the *master* video while all others are so-called *slaves*, and are basically controlled by the master video's progression in time. As opposed to ELAN, ANVIL has no facilities for changing video synchronization (usually done by defining a fixed offset between two videos).

## 4.2 Motion Capture Data

As motion capture is becoming more affordable (e.g. through the use of inertial sensors [RLS08]), such technology is becoming more likely to be employed in human behavior analysis [FvdKRvW08]. Similar technologies like cyber-gloves have already been used in sign language research [CSAW06]. In psycholinguistics, such data could bring long-awaited refinement of theories of human gesture behavior [KvGvdH98]. In a more general context, large libraries of motion capture data like the one residing at CMU Graphics Lab<sup>4</sup> will need powerful yet intuitive retrieval systems like the one in [MRC05]. For all these research issues, an annotation tool with an integrated 3D viewer would be an important asset to perform qualitative analysis or create training material.

Motion capture recording usually takes place in a studio and involves several high-speed cameras while the human performer is equipped with either passive or active markers. These markers are then used in post-processing to reconstruct the relative angles of bones with respect to joints. Fortunately, while there are multiple ways and technologies to perform motion capture, for the final representation of motion capture data there are standardized file formats. The most popular ones are Acclaim's ASF/AMC, Biovision's BVH, and the recent COLLADA format. The latter is deemed to become the new industry standard and is XML-based. All formats store two principal components: (1) the skeleton, i.e. the names of joints and their relative position/orientation toward each other, and (2) the motion data, usually represented frame-by-frame, giving angles of all joints for each frame. In ASF/AMC format, the skeleton is defined in the ASF file, and the motion data in the AMC file. In BVH and COLLADA, both is contained in a single file.

The ANVIL motion capture viewer is implemented in Java3D and can currently only read BVH files. The skeleton is read from the BVH file and transformed to a scenegraph, where each joint is modeled with a chain of scenegraph nodes that have geometry attached to it (the visual representation of a bone).

---

<sup>3</sup><http://jffmpeg.sourceforge.net>

<sup>4</sup><http://mocap.cs.cmu.edu>

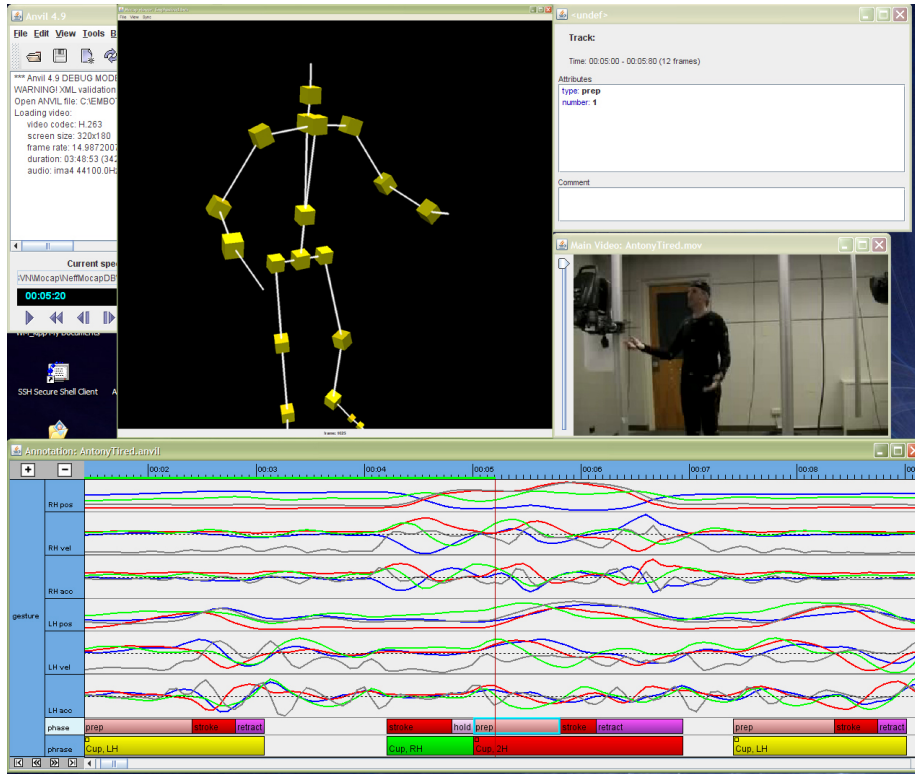


Figure 3: For viewing motion capture data, a 3D viewer can be synchronized with the regular video recording of the motion capture session.

Thanks to the scenegraph, the skeleton can be manipulated using local transforms on each joint. The motion file is stored in a separate object and the frame rate, usually around 50 fps, is sampled down to the frame rate of the corresponding video, usually 25-30 fps. The mocap viewer does not have an own clock for playback but instead listens to the signal that the master video issues each time a new video frame is displayed. This listener can be switched off for synchronization.

Synchronization between the motion capture data and the video must be done manually. The user first identifies an easy-to-recognize point in the motion capture viewer at time point  $t_m$ , then de-couples motion capture from video playback. Then, the video is brought to the equivalent point of the motion in the video, at time point  $t_v$  in the video. The two time points are then synchronized which internally means to compute their distance  $\delta = t_v - t_m$  and to use  $\delta$  as an offset when controlling the motion capture viewer.

For a first impression on how motion capture data can be useful, Figure 3 shows *motion curves* of the right and left wrist joint in 6 tracks (3 track for one wrist). The topmost track represents position in space (decomposed in x,y,z components), the next is velocity, the next acceleration. Even with bare eyes, one can see a correspondence between activity in the topmost 3 tracks (right hand wrist) and the gesture annotation in the middle of the 2 bottom tracks. Offering arbitrary motion curves in various frames of reference is subject of



future work to complement the current array of analysis feature (Section 5) which may ultimately result in an in-built classification framework similar to [MRC05].

## 5 Analysis

Analysis procedures must usually be tailored exactly to the hypotheses at hand. However, having an array of ready-to-use analysis methods in an integrated tool allows for a quick exploration of avenues one would possibly have not undertaken otherwise. Data visualization is an important tool in the analysis process and this is what transition diagrams are about. They visualize the transition behavior of categories in an intuitive fashion. As for cross-level analysis, we will present a custom process for examining the association of two attributes on different tracks that enables the user to single out the specific categories that seem to be correlated.

### 5.1 Transition Diagrams

A *transition diagram* consists of states and transitions between them (Figure 4). Each transition has a probability attached to it (in Fig. 4 the probability is measured in percent) and all outgoing transitions from a single state add up to 100% – it is therefore also a *Markov model* [PTVF07]. A transition with 21% between state *A* and state *B* means that in 21% of all times that the system was in state *A*, the immediately following state happened to be *B*. Transition diagrams visualize the temporal neighbourhood of discrete events in a quantitative fashion. For example, if looking at a stream of gestures, we may be interested in how often the speaker changes from left hand (LH) to right hand (RH) to two hands (2H), in all possible combinations (Fig. 4).

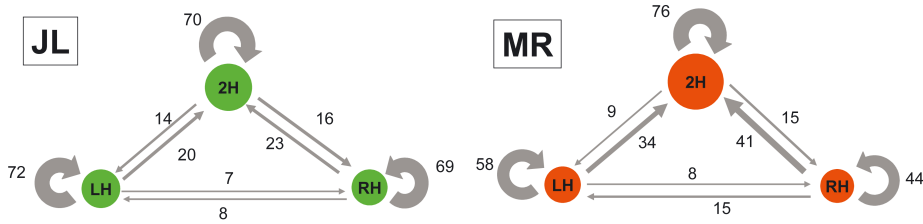


Figure 4: Handedness transition diagrams for JL and MR show preferences which hand(s) are used for gesturing and how often this mode is switched. Circle area indicates unigram probability, size of the arrows and number indicate transition probability between gestures. The diagrams show that MR uses 2H more often than JL. Moreover, JL stays in one mode more often than MR, as the high probabilities on the 2H→2H, LH→LH, and RH→RH arcs show. A switch from RH to LH and vice versa is rarely done by either speaker.

Mathematically, we model this by relative frequencies which are an approximation for the conditional probability of state *B*, e.g. LH, given that state *A*, e.g. RH, occurred beforehand. Formally, if we have a set of states  $\{s_1, \dots, s_n\}$ ,

then the conditional probability  $P(s_i|s_j)$  is approximated with the counts:

$$P(s_i|s_j) = \frac{C(s_i, s_j)}{C(s_j)}$$

where  $C(s_i, s_j)$  counts the number of occurrences of the states  $(s_i, s_j)$ , having occurred in this order, and  $C(s_i)$  counts the total number of  $s_i$  occurrences. In speech processing [JM03] this is also called a *bigram*, as opposed to the *unigram* which simply is the probability of a single state  $s_i$ , approximated by

$$P(s_i) = \frac{C(s_i)}{\sum_k C(s_k)}$$

The transition diagram as displayed in Fig. 4 is a visualization of unigrams and bigrams, where the unigram probability is indicated by the size of the circles of  $s_1, \dots, s_n$  and the bigram probabilities are indicated by the size of the arrows between the events.

The transition diagram allow to get an immediate impression of the bigram distribution and may guide the detection of regularities. The above example of gesture handedness was used, e.g. by [KNKA07, NKAS08] to detect and model idiosyncrasies in gesture behavior for two distinct speakers. The way a human speaker uses left hand, right hand or two hands is quite specific to the individual performer as recent studies support [Cal08].

## 5.2 Association Analysis

While transition diagrams give insight into the sequential behaviour of events in a single track, in association analysis we want to find out about meaningful co-occurrences of events. For example, if one track records the gesture behavior of a person and another track encodes this person’s emotional state, then we might be interested whether certain gesture types coincide with a certain emotional state. Let us assume that the interesting categories are encoded in two attributes  $A$  and  $B$  located on tracks  $T_1$  and  $T_2$  respectively (where  $T_1 \neq T_2$ ). In our example,  $A$  could be gesture handedness (LH, RH, 2H) and  $B$  could be emotional state (happy, angry ...). Since the attributes are located on different tracks, we first have to decide in which cases elements are considered to “coincide”. For each pair of coinciding elements you could then compare the values of  $A$  and  $B$ . Coincidence could be only those elements in  $T_1$  that are fully contained in an element on  $T_2$  but it could also be every pair of elements that temporally overlap. The user can formally define this using the Allen relations introduced in Section 3.3. This done, we are now able to view coinciding events in a *contingency table*.

The next step is to find out whether the two attributes are statistically associated. This is usually measured with a  $\chi^2$  test or *Cramer’s V* (a normalization of  $\chi^2$  to the interval  $[0, 1]$ ). However, this only tells us whether the *attributes* as a whole are related but not whether two specific *values* are associated. In order to find out the latter, we use an explorative method and a conclusive validation method. For the explorative part, we look at the contingency table (Figure 2). This table can be used to compute the expected value  $n_{ij}$  for each cell, defined by  $n_{ij} = \frac{N_{i.}N_{.j}}{N}$  where  $N_{i.}$  denotes the row marginals,  $N_{.j}$  the column marginals, and  $N$  the total number of observations [PTVF07].

	LH	RH	2H	$N_{.j}$
Happy	12	4	1	<i>17</i>
Angry	5	2	20	<i>27</i>
$N_{i.}$	<i>17</i>	<i>6</i>	<i>21</i>	<i>44</i>

Table 2: Example contingency table, including row and column marginals and total sum.

	LH	RH	2H
Happy	<b>6.57 (+5.43)</b>	2.32 (+1.68)	8.11 (-7.11)
Angry	10.43 (-5.43)	3.68 (-1.68)	<b>12.89 (+7.11)</b>

Table 3: Expected values matrix, including difference to actual observation.

Now the difference between expected value and actual value tells us whether there is a potential association and even the direction of this association (Table 3). However, this value is neither normalized nor is it clear what it means in terms of statistical significance. To check the hypothesis that value  $a$  out of  $A$  and value  $b$  out of  $B$  are associated, we could then run a  $\chi^2$  analysis where we treat all non- $a$  values in  $A$  as a single value  $\bar{a}$ , likewise for  $b$ . However, in order to arrive at a more precise and comparable measure of association strength, we employ the entropy-based measure of *mutual information* (MI) as suggested by [PTVF07], which is defined by

$$I(x, y) = \sum_{i,j} p_{ij} \ln \left( \frac{p_{ij}}{p_{i.} p_{.j}} \right)$$

where  $p_{ij} = \frac{N_{ij}}{N}$ . The measure is symmetrical,  $I(x, y) = I(y, x)$ , and can be used to compare the strengths of various value combinations. ANVIL displays this in an MI matrix (Table 4) which one can use to compare strengths of associations.

	LH	RH	2H
Happy	.14	.03	.25
Angry	.14	.03	.25

Table 4: Mutual information matrix.

To conclude this section, ANVIL provides a complete pipeline for finding cross-track attribute associations, including the identification and quantitative estimation of value-value associations.

### 5.3 Exporting to External Statistics Tools

Although ANVIL offers some analysis facilities, for most flexibility and power, annotation data must be exported to a format that can be read by standard statistics tools like SPSS and Statistica. Since such tools usually can read ASCII tables of some sort, this is the way export works in principle. More important is the decision of how to arrange the output data. ANVIL can output the two following structural formats:

- element-by-attribute matrix
- framenumbers-by-attribute matrix

Both outputs are tables (column separator can be defined by the user). In the element-by-attribute matrix, each row represents an annotation element (e.g. the encoding of a single gesture) while the columns contain the element's start/end time and all attributes (e.g. gesture type, handedness, shape etc.). This table may contain several tracks, resulting in a sparse table where, in each row, only those columns are filled which represent the current track's attributes. In this representation, temporal relations are neither explicit nor easy to reconstruct. Even if elements are ordered chronologically, neighboring rows need not overlap or even reside in temporal proximity. Therefore, this kind of output is problematic if one wants to analyze temporal relationships in standard statistics tools which cannot infer which elements co-occur, overlap etc. Therefore, in the second format, the framenumbers-by-attribute matrix, each row represents a single video frame like a cross-section at each time point. Columns are similar like in the first format, i.e. they represent attributes. In this table, everything represented in a row occurs at the same time, allowing for statistical analysis of co-occurrence.

Since SPSS can only handle values represented as numbers, a special option allows to export a table using numbers as codes, including a cross-reference file with the meaning of each number.

## 6 Related Work

Apart from ANVIL, a number of tools with comparable functionality exist. Most of these tools are, like ANVIL, track-based (also called tier-based), usually run on multiple platforms and output XML files. However, each tool is unique in its combination of features and usually targets a certain research community. In the following brief survey, which cannot include all the existing tools, we will point out major differences between ANVIL and the most relevant alternative tools (for more thorough tool surveys consult [BLH01, RLD<sup>+</sup>06]).

**ELAN**<sup>5</sup> is a video annotation tool developed at the MPI for Psycholinguistics [WBR<sup>+</sup>06] and bears the closest similarity to ANVIL. It is also written in Java and XML-based. Tracks are called *tiers* in ELAN. Unlike ANVIL, tiers can hold only simple elements that contain a string. A single element in ANVIL must therefore be encoded on multiple tiers in ELAN. ELAN allows for relationships between tracks: time subdivision, symbolic subdivision, included in (similar to ANVIL *span type*), and symbolic association (equivalent to ANVIL *singleton type*). ELAN also offers multiple video viewing but does not support motion capture viewing. Apart from the timeline view that ANVIL offers as the main visual interface, ELAN has several more views: a table view, a text view and an interlinear view. One major difference between ELAN and ANVIL lies in the fact that ANVIL keeps the structure of the annotation (i.e. declaration of tracks and attributes) in a separate file, the so-called *coding scheme*, whereas ELAN stores this information together with the annotated data. This can cause consistency problems when dealing with large collections of annotation files that should conform to the same scheme.

---

<sup>5</sup><http://www.lat-mpi.eu/tools/elan>

**EXMARaLDA**<sup>6</sup> is a video annotation tool mainly targeted at the research field of conversation analysis [Sch04]. It is somewhat theory-dependent (for instance, each tier has a speaker assigned to it) and based on the general annotation graph framework [BL01]. It is also Java- and XML-based, but neither supports track relationships nor complex elements.

**MacVisSTA**<sup>7</sup> is a video annotation tool targeted at human communication and interaction analysis [RQS04]. The system is restricted to Mac OS and features the integration of multiple data sources, including motion capture data. However, the latter is not displayed in the form of a 3D skeleton but only as curve plots. MacVisSTA features database integration in two ways: first to an external database for collaborative coding and second to an embedded database for querying. The hybrid architecture may be extended through plugins.

**PRAAT**<sup>8</sup> is an audio analysis and annotation tool, mainly targeted at phonetics research, developed at the Institute of Phonetic Sciences, University of Amsterdam [BW05]. It runs on multiple platforms and is certainly the most widely used tool in phonetics. For annotation, PRAAT also offers multiple tracks which come in two flavors: one records elements with a duration, one only elements with a single time point. The actual information stored in elements are simple strings. Since PRAAT allows very precise playback control on audio files, it is very suitable for speech transcription. ANVIL can import PRAAT encoded data and we actually recommend PRAAT as a supplementary tool for ANVIL to do both speech transcription and intonation analysis which can also be imported and displayed in ANVIL.

**The Transformer**<sup>9</sup> is a tool mainly targeted at social scientists that allows to convert and view files from various tools, including PRAAT and ELAN. The ANVIL format is not supported. It offers different views and embeds PRAAT as a viewer. Internally, the tool is based on a database.

**Advène**<sup>10</sup> is developed at LIRIS laboratory, University Claude Bernard Lyon 1. It aims at providing a system for sharing annotations on digital videos (movies, courses, etc) and providing tools for editing and visualization of so-called *hypervideos* which are generated from annotations and videos. Users can then exchange analyzed and commented multimedia data. Advène can import ANVIL data. Transformer and Advène can be considered meta-tools as they provide services on top of other tools, thus enabling to profit from the strengths of various tools in an integrated workflow.

To conclude, ANVIL remains the only tool that offers structured objects as annotation elements (i.e. typed attributes) instead of simple strings and, surprisingly, the only tool that keeps the coding scheme as a strictly separate file which has proven to be a major advantage in developing coding schemes. Of the reviewed tools here, it is the only tool that allows the encoding of spatial information on the video frame and the first tool to integrate a 3D motion capture viewer. Our tool shares with MacVisSTA an embedded database for complex queries and with ELAN the use of track relationships to make coding more robust. There are tools that can be used for importing to ANVIL, namely PRAAT, and tools that consume ANVIL files, namely Advène. An important

---

<sup>6</sup><http://www.exmaralda.org>

<sup>7</sup><http://sourceforge.net/projects/macvissta>

<sup>8</sup><http://www.praat.org>

<sup>9</sup><http://www.oliverehmer.de/transformer>

<sup>10</sup><http://liris.cnrs.fr/advene/index.html>

area of divergence is the video playback policy. ANVIL is restricted to JMF and JFFMPEG which introduces limitations in terms of codec choice. Tools like ELAN and MacVisSTA integrate alternative or additional libraries for video playback. A path that ANVIL will likely also go in the future. Another feature for future consideration, supported by e.g. ELAN and Transformer, are multiple alternative views (timeline, text, table) on the same annotation data.

## 7 Conclusions

We presented extensions to the ANVIL video annotation tool that are aiming at making it an integrated platform for the annotation, browsing and analysis of multimedia data. To this end, we integrated a 3D motion capture viewer and an SQL database. For the analysis across tracks the database forms the necessary basis for association detection. The association analysis uses contingency tables for identifying possible associations between attribute values and then gives *mutual information* measures to estimate the strength of these associations. We also integrated a visualization of category transitions in the form of transition diagrams. This collection of features is quite unique in the annotation tools landscape. ANVIL can now move toward including automated extraction like motion detection, either by working on motion capture data [MRC05] or by applying computer vision algorithms on the video files to perform semi-automatic annotation, ideally in a realtime, interactive process with the coder in the loop. Such directions have the potential to build new alliances between empirical researchers and information extraction communities.

On a higher level, there are two burning issues for future exploration: tool interoperability and scheme standardization. Since many annotation tools now exist, each with their own strengths and special features, it would be highly desirable to establish mechanisms that allow the joint use of several tools in a smooth workflow. Usually this implies some data transformation, which can be done with a tool like Transformer<sup>11</sup>, or a direct import feature, e.g. ANVIL users usually do their speech transcription in PRAAT and then import this data into an ANVIL track. However, given  $N$  tools one would need  $N \times N$  specific transformation procedures. Instead, if a single exchange format  $X$  existed, this could be reduced to  $N + N$  transformation procedures (export to  $X$ , import from  $X$ ). This avenue has been explored at a 2007 workshop on multimodal annotation tools [SDE<sup>+</sup>08] and resulted in a preliminary exchange format based on annotation graphs [BL01]. However, a number of important features, e.g. track relationships, are nontrivial to map, so that for now, such transformations are not lossless. The second issue is that of standardization which has been explored in [BKMW05]. The idea is to have standard coding schemes that could be manifest in coding scheme files. Along these lines, a decomposable coding scheme in the form of a meta-scheme needs to be developed. For standardization to be feasible, such meta-scheme would have to be also interoperable across many tools. This avenue seems possible since many coders (re-)use similar schemes (e.g. [KvGvdH98] for movement phases) or are connected in networks with a standardized coding procedure (e.g. the MUMIN network [ACD<sup>+</sup>05]).

---

<sup>11</sup><http://www.oliverehmer.de/transformer>

## Acknowledgements

Special thanks to Quan Nguyen and Gabriel Manolache for their programming work on export functions and database integration. Thanks to Nele Dael, Marcello Mortillaro and Klaus Scherer (U Geneva, CISA) for suggesting the frame-by-frame export. This research has been carried out within the framework of the Cluster of Excellence *Multimodal Computing and Interaction* (MMCI), sponsored by the German Research Foundation (DFG).

## References

- [ACD<sup>+</sup>05] Jens Allwood, Loredana Cerrato, Laila Dybkjaer, Kristiina Jokinen, Costanza Navarretta, and Patrizia Paggio. The MUMIN multimodal coding scheme. Technical report, 2005.
- [BKMW05] Harry Bunt, Michael Kipp, Mark T. Maybury, and Wolfgang Wahlster. Fusion and Coordination For Multimodal Interactive Information Presentation. In Oliviero Stock and Massimo Zancanaro, editors, *Multimodal Intelligent Information Presentation*. Springer, 2005.
- [BL01] Steven Bird and Mark Liberman. A Formal Framework for Linguistic Annotation. *Speech Communication*, 33(1–2):23–60, 2001.
- [BLH01] Tony Bigbee, Dan Loehr, and Lisa Harper. Emerging requirements for multi-modal annotation and analysis tools. In *Proceedings of the 7th European Conference on Speech Communication and Technology (Eurospeech)*, pages 1533–1536, 2001.
- [BSP<sup>+</sup>04] J. Barbic, A. Safonova, J.-Y. Pan, C. Faloutsos, J. K. Hodgins, and N. S. Pollard. Segmenting motion capture data into distinct behaviors. In *Proc. of Int. Conf. on Graphics interface*, pages 185–194, School of Computer Science, University of Waterloo, Waterloo, Ontario, Canada, 2004. Canadian Human-Computer Communications Society.
- [BW05] Paul Boersma and David Weenink. Praat: doing phonetics by computer (version 4.3.14) [computer program]. Retrieved from <http://www.praat.org/>, 2005.
- [Cal08] Geneviève Calbris. From left to right...: Coverbal gestures and their symbolic use of space. In *Metaphor and Gesture*, pages 27–53. John Benjamins, 2008.
- [CSAW06] Onno Crasborn, Han Sloetjes, Eric Auer, and Peter Wittenburg. Combining video and numeric data in the analysis of sign languages within the elan annotation software. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC)*, 2006.

- [FvdKRvW08] Wim Fikkert, Herman van der Kooij, Zsofia Ruttkay, and Herwin van Welbergen. Measuring behavior using motion capture symposium. *Proceedings of Measuring Behavior 2008*, August 2008.
- [JM03] Daniel Jurafsky and James H. Martin. *Speech and Language Processing*. Prentice Hall, 2003.
- [Kip01] Michael Kipp. Anvil – a Generic Annotation Tool for Multimodal Dialogue. In *Proceedings of Eurospeech*, pages 1367–1370, 2001.
- [Kip08] Michael Kipp. Spatiotemporal coding in anvil. In *Proceedings of LREC*, 2008.
- [KNA07] Michael Kipp, Michael Neff, and Irene Albrecht. An Annotation Scheme for Conversational Gestures: How to economically capture timing and form. *Journal on Language Resources and Evaluation - Special Issue on Multimodal Corpora*, 41(3-4):325–339, December 2007.
- [KNKA07] Michael Kipp, Michael Neff, Kerstin H. Kipp, and Irene Albrecht. Toward Natural Gesture Synthesis: Evaluating gesture units in a data-driven approach. In *Proc. of the 7th International Conference on Intelligent Virtual Agents (IVA-07)*, pages 15–28. Springer, 2007.
- [KvGvdH98] Sotaro Kita, Ingeborg van Gijn, and Harry van der Hulst. Movement phases in signs and co-speech gestures, and their transcription by human coders. In Ipke Wachsmuth and Martin Fröhlich, editors, *Gesture and Sign Language in Human-Computer Interaction*, pages 23–35, Berlin, 1998. Springer.
- [MK02] Jean-Claude Martin and Michael Kipp. Annotating and Measuring Multimodal Behaviour – Tycoon Metrics in the Anvil Tool. In *Proceedings of the Third International Conference on Language Resources and Evaluation (LREC)*, pages 31–35, 2002.
- [MRC05] Meinard Müller, Tido Röder, and Michael Clausen. Efficient content-based retrieval of motion capture data. *ACM Trans. Graph.*, 24(3):677–685, 2005.
- [NKAS08] Michael Neff, Michael Kipp, Irene Albrecht, and Hans-Peter Seidel. Gesture Modeling and Animation Based on a Probabilistic Recreation of Speaker Style. *ACM Transactions on Graphics*, 27(1):1–24, March 2008.
- [PTVF07] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical Recipes: The Art of Scientific Computing*. Cambridge University Press, third edition, 2007.
- [RLD<sup>+</sup>06] Katharina Rohlfing, Daniel Loehr, Susan Duncan, Amanda Brown, Amy Franklin, Irene Kimbara, Jan-Torsten Milde, Fey Parrill, Travis Rose, Thomas Schmidt, Han Sloetjes, Alexandra



- Thies, and Sandra Wellinghoff. Comparison of multimodal annotation tools — workshop report. *Gesprächsforschung*, 7:99–123, 2006. EN.
- [RLS08] Daniel Roetenberg, Henk Luinge, and Per Slycke. 6 dof motion analysis using inertial sensors. In *Proceedings of Measuring Behavior 2008*, 2008.
- [RQS04] T. Rose, F. Quek, and Y. Shi. Macvissta: A system for multimodal analysis. In *Proceedings of the 6th International Conference on Multimodal Interfaces*, 2004.
- [Sch04] Thomas Schmidt. Transcribing and annotating spoken language with exmaralda. In *Proceedings of the LREC-Workshop on XML based richly annotated corpora, Lisbon 2004*, Paris, 2004.
- [SDE<sup>+</sup>08] Thomas Schmidt, Susan Duncan, Oliver Ehmer, Jeffrey Hoyt, Michael Kipp, Dan Loehr, Magnus Magnusson, Travis Rose, and Han Sloetjes. An exchange format for multimodal annotations. In *Proceedings of LREC*, 2008.
- [WBR<sup>+</sup>06] Peter Wittenburg, Hennie Brugman, Albert Russel, Alex Klassmann, and Han Sloetjes. Elan: a professional framework for multimodality research. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC)*, 2006.