Application of Binary Search to Video Annotation and Behavior Tracking for the Social Sciences

Caitlyn Lee
Dartmouth College

Follow this and additional works at: https://digitalcommons.dartmouth.edu/senior_theses

Part of the Computer Sciences Commons

Recommended Citation
Lee, Caitlyn, "Application of Binary Search to Video Annotation and Behavior Tracking for the Social Sciences" (2019). Dartmouth College Undergraduate Theses. 145.
https://digitalcommons.dartmouth.edu/senior_theses/145

This Thesis (Undergraduate) is brought to you for free and open access by the Theses and Dissertations at Dartmouth Digital Commons. It has been accepted for inclusion in Dartmouth College Undergraduate Theses by an authorized administrator of Dartmouth Digital Commons. For more information, please contact dartmouthdigitalcommons@groups.dartmouth.edu.
Application of Binary Search to Video Annotation and Behavior Tracking for the Social Sciences

By
Caitlyn Lee

Computer Science Honors Thesis
May 30, 2019
Department of Computer Science
Dartmouth College

Advisor: Lorie Loeb, Ph.D.
Second Reader: Xing-Dong Yang, Ph.D.
Third Reader: Soroush Vosoughi, Ph.D.
# Table of Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abstract</td>
<td>3</td>
</tr>
<tr>
<td>Introduction</td>
<td>4</td>
</tr>
<tr>
<td>Related Work</td>
<td>5</td>
</tr>
<tr>
<td>Implementation</td>
<td>8</td>
</tr>
<tr>
<td>Case Study</td>
<td>10</td>
</tr>
<tr>
<td>User Interface</td>
<td>15</td>
</tr>
<tr>
<td>Analysis</td>
<td>21</td>
</tr>
<tr>
<td>Discussion</td>
<td>26</td>
</tr>
<tr>
<td>References</td>
<td>28</td>
</tr>
</tbody>
</table>
Abstract

Annotation and labeling is a critical component of computer vision. However, completed manually, this process is time and cost-intensive. In particular, video annotation is particularly arduous in terms of manual annotation and therefore is additionally costly. Because videos are often annotated frame by frame, making little use of the fact that the data between any two consecutive frames are closely related, the process of completing a single video annotation is the equivalent of the cumulative work of annotating and labeling an equal number of distinct images. For certain applications of video annotation, we can leverage assumptions about the objects in the video that allows us to most efficiently utilize the similarity between the frames of a video. In this paper, I describe an analysis of a new software that implements a binary search algorithm for the use in video annotation for the social sciences. I present a specific case study that describes and analyzes the usage and efficacy of this software for tracking individuals in a social gathering for psychological research and discuss how the software may be used in other similar applications in the future.
Introduction

Annotation and labeling is a critical component of computer vision. However, unlike image labeling which can be crowdsourced and completed efficiently, both in terms of cost and accuracy, video annotating is a much harder process. Because videos must be annotated frame by frame, the process of completing a single video annotation is often thought of as comparable to the sum of labeling each of the frames of the video as single images. This process is often extraordinarily inefficient and repetitive. The current best video annotation tools require the user to step chronologically through the video annotating for any object or action at each given frame. The similarity between two contiguous frames is typically only leveraged insofar as an object’s location annotated on one frame will remain in that same location in the next frame. Some tools also allow the ability to fast-forward and rewind through the video at different paces, as well as the ability to jump through frames to try and make the process more efficient. Even still, some “data sets consisting of millions of frames have cost tens of thousands of dollars and required up to a year of continuous work to annotate” (Vondrick, Patterson & Ramanan, 2012). In addition, many video annotating tools and software are more difficult to use than simple image labeling tools, and thus require greater investment in initial training.

After reviewing the currently available annotation tools and studies of video annotation as well as the specific contexts for use, I propose in this paper a simple implementation of binary search which makes video annotation, specifically for the use in the behavioral and social sciences, much more efficient and easy to use while maintaining a high standard of quality in the final annotations.
Related work

The problem of image labeling has many low-cost efficient solutions that has led to the completion of many large-scale image data sets such as ImageNet, PASCAL, SUN, and TinyImages. Building off of these image data sets, the computer vision community has developed a handful of visual annotation tools.

LabelMe

LabelMe is an online platform where users can upload their own images and create polygonal paths to generate object and event annotations. LabelMe applies the principle of label imaging where the object can be specifically isolated from the other objects, such as the trees and the cannon below in Figure 1a and 1b, respectively. However, if we are annotating objects whose shape might change between frames such as people, animals, or even moving cars, then we must re-create the specific polygonal path to outline the shape every time the distinct shape of the object changes. LabelMe, however, is very good at creating “complex event annotations between interacting objects” and building large data sets of high quality.

Fig 1. a) A screenshot from LabelMe showing the objects that have been labeled in the sample image. Each object labeled is shown in a distinct color (the trees, the bus, the buildings, etc.). b) Another screenshot from LabelMe showing the polygonal path that the user creates to identify objects in the image
Similar such tools to LabelMe include ViPER (Mihalick and Doermann, 2003), FlowBoost (Ali et al., 2011) and TRECVID (Smeaton et al., 2006). These tools involve the annotating many images at a very detailed level to create large data sets. Each of these tools focuses on a specific aspect of the video annotation problem that makes the tool well-suited to certain applications. For example, ViPER is specifically optimized for spatial labeling while FlowBoost uses sparse sets of key frame annotations.

*Loopy*

Another web-based video annotation tool called Loopy tries to be as general as possible, without targeting a specific facet of the video annotation challenge to be as robust and versatile as possible. Users upload their videos as videos as opposed to single frames, create annotation classes and custom attributes, and then annotate the video using the defined annotation classes. The focus of the objects identified in the video with Loopy is their location, as the actual objects are simply denoted using a resizable rectangle. The user progresses through the video and moves the rectangular objects to update the location of the individuals. Without manual corrections, the program leaves the identified objects in the same place between frames.
Fig 2. a) A screenshot from Loopy. The objects in this image belong to the annotation class “person”. The “person” annotated on the right, in the yellow, has moved from the last frame where they were annotated and the annotation is no longer correct.

Because Loopy is focused on the location of the object as opposed to the specific shape of the object as in LabelMe above, the “correction” that must take place in a future frame if the old annotation is no longer correct is much more efficient. The only property of the old annotation that has to change is the location, whereas in LabelMe and other image-labeling based video annotation tools, the object itself that has been identified often needs to be edited or entirely redone.
Another challenge with all the aforementioned video annotation tools is the cost, both in terms of financial cost and the manual labor that is required to do the annotation. The tools are all capable of creating high quality large data sets, it is not financially feasible for annotating many videos. The tools also overlook the willingness and ability of the users who are doing the annotation; they must be motivated and willing to complete the entire process of the annotation, otherwise the manual labor cost is prohibitively high. In addition, the more training any given tool requires before it can be successfully used, the less widespread and community-oriented it can be in terms of usage.

**Implementation of Binary Search**

For certain applications, especially those pertaining to the social sciences, the objects in the video may move following a more narrowly defined set of assumptions. For example, in videos of naturalistic interactions between humans (a busy street corner, a park, a party, etc.), these assumptions may be that the people are not moving through the entire duration of the video and when they are moving, they tend to move at a constant, if not fixed, rate. As such, in developing a tool for video annotation catered for the social sciences, we can utilize these assumptions and eliminate unnecessary features while streamlining the entire annotation process.

If we accept and operate under the two main assumptions that a) the objects (persons) in the video are mainly in a fixed location while they are interacting with other individuals and b) when the objects do move, they do so at a constant rate (they walk from one point to another as opposed to beginning with a walk that turns to a run). Under these assumptions, the problem of tracking individual objects’ movements over time can then be reframed as identifying the moments that an individual is moving versus standing still. More explicitly, the tracking can be
thought of as partitioning all frames or time points, where a partition is defined as a continuous period of time where the object is either moving or standing still. At each time point $t_i$ the location of the individual is stored.

Because we are assuming the objects, when in motion, are moving at a constant rate, if we know that an object is moving between time points $t_1$ and $t_2$, then we can use a simple linear interpolation between the locations at any two time points $t_i$ and $t_j$, $t_1 \leq t_i < t_j \leq t_2$ to get the locations at every other time point $t_k$, $t_i \leq t_k \leq t_j$. Trivially, if we know that an object is standing still between any two time points $t_1$ and $t_2$, then we can use the same linear interpolation idea as if the object were moving. The object’s location will not have changed between time points $t_i$ and $t_j$ and thus the linear interpolation would always, correctly, return the same location. So then, we can say that given the locations of an object at two time points $t_1$ and $t_2$, if we know that the time points belong to the same partition then we can linearly interpolate between them to get the intermediary positions without needing to manually annotate them all.

Furthermore, we can see that if we are given two time points $t_1$ and $t_2$ and we compute the linear interpolation between the locations at those two points, if $t_1$ and $t_2$ are in the same partition, then the locations at every intermediary time step must be accurate. If any such intermediary location is not accurate, then it must be true that $t_1$ and $t_2$ are in different partitions.

If we know the time points that define the partitions in the video we are annotating, then from those time points we can linearly interpolate to get the location of the object at every time point in the video. These partition boundaries can be found using a modified binary search algorithm. Given locations at any two time points $t_i$ and $t_j$, we can linearly interpolate to calculate the location at the midpoint $t_m, m = \text{ceil} \left( \frac{t_i + t_j}{2} \right)$. At this midpoint, we can verify whether or not the calculated location of the object is correct. If correct, then we know that $t_i$ and
$t_j$ are in the same partition and we continue. If the calculated location is incorrect, then we know that $t_i$ and $t_j$ must not be in the same partition and there exists at least one partition boundary between $t_i$ and $t_j$. We linearly interpolate between now the known locations at time points $t_i$ and $t_m$ and time points $t_m$ and $t_f$. This process continues recursively until we have found all of the partition boundaries; until between every pair of known locations, the midway linear interpolation is correct.

The implementation of the algorithm is as follows. The user begins by manually annotating the location of the object in the first frame and the last frame. Then, using linear interpolation, the midpoint coordinate between the locations is computed and shown to the user. If the object is not in the expected location at that middle time point, then the user is asked to correct the location, and the recursive midpoint checking continues between the next set of known locations. This process continues until the recursive stack is empty and the location of the object at every time point can be calculated from the known locations and time points.

**Case Study**

*Background*

Despite the fact that social interaction is one of the most ubiquitous human behaviors, we know very little about how social interaction works or why it is protective for our physical and mental health. In large part, this gap in our knowledge exists because the fields of psychology and neuroscience rely on laboratory studies, most of which test participants in isolation (Wheatley, Boncz, Toni, Stolk, in press). The paradigms, equipment, and analytical tools available to psychologists and neuroscientists have been designed primarily to study one mind at a time. With new computational tools, this constraint is weakening.
As a specific case study and test of the algorithm described above, I implemented the algorithm into a standalone video annotation software that was used to annotate the entire video for a psychology study analyzing the social network and conversational dynamics of an environment with large-scale naturalistic social interactions. As such, not only were the locations of the objects (individuals at the party) being tracked, but also the conversations that they were a part of.

Study

The purpose of the study was to observe people at a social mixer where the individuals were largely unfamiliar with one another at the beginning of the event: a mixer for the first-year Tuck Business school students. The Tuck Business school population is relatively unique in certain social aspects. They are somewhat isolated from other people their age and everyone’s social groups are almost entirely comprised of other Tuck students. The students all take courses together, often live together, and spend their free time with one another as well. This closeness between the students combined with their isolation from others outside of the business school makes their social network exist in almost a vacuum. Because their interactions are largely confined to a few specific social settings, we may be able to predict features of the social network simply from the behavior of the individuals. The traits that may make someone a hub in the social network may be visible in their interactions with their classmates in social settings. We observed their movement patterns as well as their conversational behaviors using hidden cameras placed along the ceiling of the room where the mixer was being held. We annotated the video using the new software mentioned above, and began to explore some potential analyses that can be done using these data.
Before now, however, the process of quantifying these large-scale naturalistic behaviors proved challenging enough in and of itself that these types of studies were often overlooked for their laboratory counterparts.

*Users*

The users for this case study were primarily psychologists and research assistants working in psychology labs. The stakeholders for this study were the lead researchers. These are users whose main intentions were not to build massive datasets to train deep models, but to most efficiently and easily acquire the data that they needed for their respective studies. They wanted data that was accurate enough to provide relevant information, while still being collected quickly and efficiently enough to be cost-effective. The actual users for the software, the individuals that would be completing the annotations were the lab and research assistants. The research assistant’s main goals were to have tools that were easy to use. Furthermore, many of the research assistants did not have any technical or engineering experience, so many of the existing tools that required complex installation processes provided a significant barrier to usage. They also wanted the final output to be both visual and in forms that could be used for subsequent analyses.

From user interviews, the researchers indicated that the existing software for video was often too feature-heavy to be efficient. They didn’t need the ability to annotate specific movements or have multiple classes of objects they were annotating for.

In addition, much of the existing and open-source software that was available for video annotation was designed for short clips to use as training data for action-identification. But because the end goal for these annotations was to be used as large-scale data, many of the software options were not well-suited to handle large video files.
Video specifications

The specific video that was being used for this case study was footage of a single party. The cameras were at the edges of the ceiling and looking down on the room. At any given moment, the number of people in the room ranged from 4 to 83. The “difficulty” of the video annotation varied depending on the number of people in the room, the amount of occlusion, and the amount of movement of the individuals that are being tracked. Sample frames from the video are shown below.

Fig 3. a) (above) A sample frame that contains a large number of individuals and a significant amount of occlusion for any given individual. b) (below) A sample frame that contains a moderate number of individuals. The individuals are spread out in such a manner that the amount of occlusion is low. c) (next page) A sample frame from the end of the party with few individuals. The amount of occlusion is minimal.
The video was cut into frames at a rate of 1FPS, for a total of 710 frames for each 11 minute and 50 second clip. There was a total of 14 clips, marking the entire duration of the party.
Each clip was annotated by a different user – where annotating the entire clip entailed annotating the locations of every individual in the room for the whole clip.

**User Interface of Software**

The software was all written in Python, originally created in interactive python notebooks. The notebooks were given to each user fully pre-packaged and ready to run. The users simply needed to run the appropriate cells in order to run the software and do the annotations. The annotation process was as follows.

The user could specify the ‘start’ and ‘stop’ frames that they wanted to annotate between. If an individual was only in the frame for a short amount of time they had the option to specify that as a parameter. Otherwise, the default ‘start’ and ‘stop’ frames were the first and last frames of the video, respectively. After specifying these parameters, the user was shown the first frame of the video and asked to identify the given individual they were trying to annotate for.
Fig 4. The first annotation frame. The red circle indicates where the user has indicated the location of the object of the tracking to be.

Then, the user was shown the last specified frame and asked to identify where the same individual was on the frame. The user had the ability, at this moment or any subsequent time in the future, to go back to the first frame to reference the individual they were trying to track.
Fig 5. The last annotation frame. The red circle indicates where the user has indicated the location of the object of the tracking to be.

The user was next shown the first midpoint frame, along with a guess according to the linear interpolation as to where the individual is expected to be. If this guess is correct, then the user simply continues on and if there are frames still to check on the recursive stack, they continue to check and correct until the stack is empty. Once the stack is empty, the program automatically exits and the data is saved.
Fig 6. Sample midpoint frames showing guesses as to where the individual might be according to the linear interpolation.
On each iteration, the individuals that have already been tracked are shown to the user when they are shown the first frame:

Fig 6. On the next run of the program the first frame indicates the user that has already been tracked.

Once all the videos were tracked, the locations of the individuals were overlayed onto the video frames to show the positions of all the individuals. Sample frames are shown below.
Fig 7. Sample finished frames.

Analysis of software and algorithm

When the annotations were completed, there was a total of 1510 individual annotations completed, where each annotation consisted of a single individual being tracked across a 710-frame video clip set. As described above, the number of partitions for a certain individual’s locations across the time-series represents the number of times they start and stop moving. For each annotation, the number of “clicks” or edits that the user made during the tracking process was recorded. A single click represented an instance of a midpoint location being incorrect, where the user needed to update the location of the individual, thus adding two more midpoint frames to check on the recursive stack.

We found that the number of edits was closely correlated with the number of partitions of a certain individual’s movements ($r = 0.900, p < 0.001$).
The total number of edits varies directly with the total number of partitions in an individual’s movement patterns. The total number of edits per partition is about 2.5.

The slope of the line is 2.5, signifying that for every change in movement type (between standing still and walking around), the user needed to make on average 2.5 clicks to find the time point at which that change occurred. Compared to traditional video annotation tools where the user would need to scrub through the video to manually identify the moments at which an object may start or stop moving, using this software the users were able to find such moments in less than 3 clicks, with no a priori knowledge as to the movement patterns of the object.

One of the aforementioned ways to measure difficulty in tracking an object was the amount that the object moved during the video. One such way to measure the amount of movement is in the number of times the object starts and stops (the number of partitions), and another way is to measure the total distance that the object covers over the course of the video.

For our case study, the number of clicks was also strongly correlated with the amount of distance
that the object moved over the course of the 710 frames ($r = 0.786, p < 0.001$).

Fig 9. The number of edits increases slightly as the amount of distance traveled increases.

The slope of the line of best fit however, is only 0.02, indicating that as the amount of
distance traveled increases, the number of clicks increases significantly slower. Within two
standard deviations of the mean of the amount of distance moved by the individuals in the
frames, the increase in the number of clicks is minimal. This demonstrates that even as the task
becomes significantly more difficult, the process of annotating the individuals increases in
difficulty at a much slower rate.
Fig 10. Curves showing the proportion of total individuals that can be completely tracked in a 710-frame segment using a fixed number of clips.

The curves above demonstrate, using a fixed number of clicks, the proportion of individuals that move a certain amount at the party that can be tracked entirely across a 710-frame segment. Everyone, regardless of how much they move, can be tracked across the 710-frame segment by clicking on only 36% of the total number of frames. And using only 50 clicks, a user is still able
to track more than 60 percent of individuals that cover the average amount of distance that any individual moves during the party.

The final measure of difficulty for doing the location tracking was the amount of occlusion present in any given frame. To quantify the amount of occlusion, for each person in the frame, the number of neighbors within a certain threshold were counted. The threshold that was used was 1.5 times the size of the radius of the ellipse that was used in place of a bounding box to track the individuals. Any neighbor within this radius would mean that the ellipse used to track them must overlap, at least partially, with the current individual. There was almost no relationship between the total number of edits and the total number of neighbors that an individual had over the course of a single 710 clip.
**Discussion**

The tradeoff between accuracy and efficiency was one of the key challenges in designing a new video annotation tool. The more accurate and precise a user wanted to be, the longer and more arduous the task of completing the annotation. In the software that I created for this task, I built accuracy in as a user-defined feature. For the case study described above, we chose to splice the video into frames at a rate of 1FPS, because for the location data over the course of two hours, any scale finer than that would be unnecessary. Within the actual annotations, the users were told to be as accurate as possible, trying to center the ellipse indicating the location of the individual around the individual’s head. Then, we measured to see if we could still maximize efficiency given these specified parameter.

According to the analysis above, the program did quite well in being efficient and robust to the variability in the difficulty of the clips. Even as the difficulty of the annotation increased, in terms of the amount of movement of a given individual and the amount that the individual was occluded, the efficiency of the program remained relatively constant, varying only in response to the target variable: the number of times that the individual starts and stops moving.

Instead of traditional video annotation tools where the user must scrub through the entire video chronologically at a predefined or variable rate, this software allows for the user to look at close to the minimal number of frames to find the points where the objects are beginning to and stopping moving. Even at increased rates, other video annotation tools require the user to look at every frame of the video, regardless of the fact that in most frames the individual was not moving.

The main drawback to the approach presented in this paper is the propensity to lose sight of an individual across large gaps of time. Despite the ability to toggle back and forth from the
initial frame to check the identity of the individual being tracked, the algorithm does not account for the possible directions they could be moving or where they are going, including potentially out of the frame. While because in this setting, the room was small enough and the camera angles detailed enough that the individuals could be clearly and easily identified as unique, individuals were rarely “lost” and later identified as a different individual, it is easy to see other settings in which this would present a greater challenge. For example, in looking at grainy low-resolution street camera footage, if an individual were to walk outside the frame and re-enter not in the exact same location, it would be nearly impossible to tell that it was the same individual. Similarly, if it were a crowded street corner and the person bent down to pick something up or quickly walked to the other side of the street and back, across a large swath of time, simply looking at interspersed frames would not be helpful in identifying where the individual had gone, if we only know that they are moving somewhere in the space.

However, despite this challenge, the application of binary search within a video annotation tool has been shown in this paper to be significantly more efficient when compared to existing video annotation software, particularly pertaining to applications within the social sciences. Because of certain assumptions we can make about the objects of study within those domains, we are able to utilize the binary search algorithm to reframe the video annotation problem as a search problem for moments of movement, rather than identifying the action or inaction at every specific time point.
References


