Planning Coordinated Event Observation for Structured Narratives

Dylan A. Shell[1], Li Huang[2], Aaron T. Becker[2], Jason M. O'Kane[3]

***Abstract*— This paper addresses the problem of using autonomous robots to record events that obey narrative structure. The work is motivated by a vision of robot teams that can, for example, produce individualized highlight videos for each runner in a large-scale road race such as a marathon. We introduce a method for specifying the desired structure as a function that describes how well the captured events can be used to produce an output that meets the specification. This function is specified in a compact, legible form similar to a weighted finite automaton. Then we describe a planner that uses simple predictions of future events to coordinate the robots' efforts to capture the most important events, as determined by the specification. We describe an implementation of this approach, and demonstrate its effectiveness in a simulated race scenario both in simulation and in a hardware testbed.**

I. Introduction

What if you asked your robot to tell a story? Generating narratives that capture the salient structure of a series of events is a common skill in the daily life of humans. This paper begins to explore the question of how to direct robots to achieve such things autonomously. Though robots capture substantial quantities of video data, video captured by robots has thus far largely lacked narrative structure. In this work, we explore automatic video narrative generation by a coordinated team of robots equipped with cameras.

Our interest in this sort of narrative capture by robot teams is motivated by a number of potential applications. For example, imagine tasking robot teams with capturing and assembling video that responds to requests like these:

> "Show a sequence of events that confirms that the suspect is (not) guilty."
> "Show a sequence of events that summarize my grandfather's day, without compromising his privacy."
> "Show a sequence of events from this cricket match that explains why my side lost."
> "Show a sequence of events that identify the cause of the explosion."

The common thread through each of these requests is that they can be characterized as tasks in which robots should capture video of certain types of events occurring within a domain, experienced by one or more agents, based on a specification of the desired narrative structure.

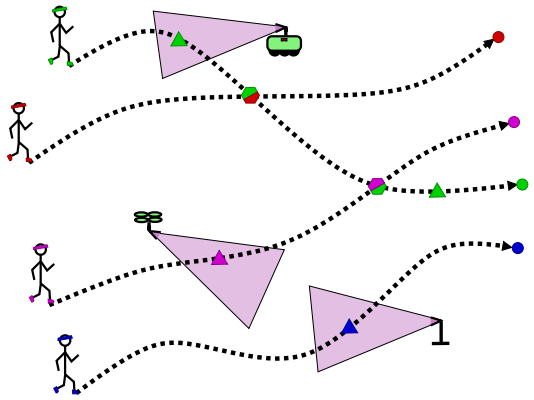

Fig. 1. System overview. Actors ( ) move through the environment, experiencing events ( ). Some of these events are captured by observers ( ), including (possibly heterogeneous) mobile robots and static cameras.

Though the technical approach we propose is suitable for a variety of these kinds of narrative capture applications, this paper aims for clarity by focusing on the example application of generating individualized highlight videos for each human runner in a long-distance road race, such as a marathon. The race application is a useful reference point because it encapsulates many of the most salient elements of the larger problem in a tractable setting.

For a sense of the current state of the art, consider the effort by race sponsor Adidas to provide such videos for runners in the 2018 Boston Marathon, based on static cameras placed in two locations (about 15 km from the start, and at the finish line). The resulting films are short (46 seconds), and are composed mostly of stock footage superimposed with race statistics of the runner. Little individualized footage is included, and the general structure of the highlight videos remains the same for every participant. For example, the highlight video for men's winner Yuki Kawauchi[1] is little different than the video for any other runner,[2] and in particular omits Kawauchi's dramatic overtaking of Geoffrey Kirui late in the race. Other participants have anecdotally reported seeing little or no footage of themselves at all in their 'personalized' highlight videos. Thus, though there is demonstrable interest in these kinds of videos, current technology seems unable to deliver videos that can capture the essential events to convey a desired narrative structure.

This work was supported in part by the NSF under Grant Nos. [IIS-1553063], [IIS-1619278], [IIS-1526862], and [IIS-1527436].

[1]D. A. Shell is with the Department of Computer Science and Engineering, Texas A&M University, College Station, TX 77843 USA dshell@tamu.edu

[2]L. Huang and A. T. Becker are with the Department of Electrical and Computer Engineering, University of Houston, Houston, TX 77204 USA {lhuang28,atbecker}@uh.edu

[3]J. M. O'Kane is with the Department of Computer Science and Engineering, University of South Carolina, Columbia, SC 29208 USA jokane@cse.sc.edu

[1]www.heretocreatelegend.adidas.com/id/E8P2D9oQpl

[2]Seen, for example, by entering an arbitrary bib number at http://www.heretocreatelegend.adidas.com.

Figure 1 is a system overview. We anticipate a number of important obstacles to meeting the challenges underlying this kind of system.

(1) *Mobility*: Statically-placed cameras, though undoubtedly helpful, are insufficient to capture events that are distributed widely over space. Instead, we envision teams of camera-equipped mobile robots that move through the environment, intelligently capturing events of interest.
(2) *Specification*: It is not enough to merely capture individual events that are noteworthy in isolation; we seek sequences of events that, when experienced together as a whole, tell a complete story. Thus, we develop an approach that enables a precise specification of individual events can be assembled into a complete narrative.
(3) *Prediction and planning*: To effectively position the robots for event capture, the system must predict when and where those events may occur in the future, and allocate robots to positions where those robots are likely to capture events that contribute to the overall narrative.
(4) *Postproduction*: After the video capture is complete, the system must assemble the captured video into a final product, selecting captured events into a coherent narrative.

Though this paper will perhaps raise more questions than it answers, it does make several specific contributions toward addressing those challenges.

(1) We introduce the active narrative capture by robot teams problem. Our formalization of the problem includes an approach for *story specification*, along with a quantitative objective function determining how well a set of captured events can be used to generate a story for such a specification.
(2) We show, via simulation, that even relatively straightforward approaches for *event prediction* and planning for *event capture* can successfully capture events that comprise a story matching such specifications. We also demonstrate, in an experimental testbed using Anki Overdrive robots, the feasibility of this approach in physical systems.

II. Related work

Efforts to model narrative in formal terms date back at least to Vladimir Propp, who endeavored in 1928 to catalog and classify extant Russian fairy tales according to the types and sequences of important events in those stories, providing a 'grammar' of sorts for that genre of literature [1]–[3]. That work serves as inspiration for ours, particularly for our proposed specification of the desired narrative structure.

Several other lines of prior work have tackled parts of the problem of automated capture and construction of narrative video. One related problem is that of *video summarization*, in which the objective is to select the most important shots or frames from an input video. A rich literature on video summarization exists, which can generally be divided into those that use hand-crafted criteria for the selection [4]–[7], and those that utilize learned criteria [8], [9]. Specific efforts for video summarization have been made, in the context of sport highlight video, for example for baseball [10] and cricket [11]. Of particular interest is that of Lu and Grauman [12], which performs story-driven summarization by selecting subshots based on the influence, measured by the presence of certain visual objects, of each shot on subsequent shots. In contrast to this prior work on video summarization, our paper commands a spatially-distributed set of robots to record video likely to include valuable events. The problems differ, because our controller must direct robots on-the-fly and therefore the system may fail to capture valuable events.

A related thread, considering the automatic generation of highlight videos, is being pursued in the video game industry [13]. In commercial practice, some video games can generate highlight films after the conclusion of a play session. However, the task in this context is even easier than in video summarization, because the game software can track all relevant events, and render the most impactful of them *a posteriori*. Artificial environments can also construct virtual cameras during post-processing to generate video that best shows the selected event.

Another similar problem is the *vacation snapshot problem*, in which a moving robot must capture a collection of k highly diverse photographs along its journey, with the constraint that the decision to keep or discard each incoming frame must be made immediately. Girdhar and Dudek [14] showed how to treat this problem as an instance of the k-centers problem using an online algorithm to collect the k extremum samples. More recently, a notion of curiosity was used to expand this idea to active decision-making about how the robot should move [15]. Rabinovich and Girdhar [16] applied this algorithm to online generation of video spoilers of films with modest success.

Yu and LaValle's research on story validation can be viewed as an inverse of the approach we propose here. That work assumes a sparse network of sensors has captured a sequence of events triggered by a mobile agent, and the problem is to determine whether that sequence of events is consistent with a hypothetical story, expressed as a sequence of events purported to have occurred. Progress has been made on this problem for both exact [17] and (more directly relevant to the present work) approximate forms [18] of matching. Several important differences exist between that prior work and ours. Perhaps the most essential is that, in Yu and LaValle, the sequence of captured events is treated as a fixed input. In contrast, our problem considers how to position the robots to capture events that correspond to given input story specification. This same contrast distinguishes our problem from work on Robocup commentary [19], [20], as well as the vast and active community of researchers [21]–[29] applying computational and classical planning techniques to generate narratives and text to tell stories.

III. Problem statement

This section introduces and formalizes our narrative capture problem. The central elements of the model are *actors*, *events*, and *observers*.

A. Actors

A collection of *actors* $A = a_1, \ldots, a_n$ move independently through an environment, which we denote W. The actors are the agents whose activities constitute the narratives we want to capture.

Example 1: In the road race scenario introduced in the introduction, each runner in the race is an actor. We might model the race course, from start to finish, as a continuous one-dimensional interval, so that the location of each actor at any point is determined by the fraction of the race that runner has completed.

Example 2: Another example scenario is a kindergarten recital, in which a large number of five-year-olds perform adorable music, but remain in essentially static positions. In this case, each of the children is an actor, and the environment might be modeled as a 2-dimensional region defined by the shape of the stage.

Example 3: Suppose we want to monitor the activities of pedestrians in a crowded marketplace. We would treat each person in the scene as an actor. In this case, the pedestrians' business may take them along various routes, and thus their positions within the environment will be at least partially unpredictable.

B. Events

The objective of our system is to capture (some of) the activities of the actors. We abstract these activities as a sequence of discrete *events*. Each individual event is associated with one or more actors, and occurs at some specific time and location within W.

Example 4: Continuing from Example 1, events of interest in the race scenario may include when each runner passes key locations along the race course, such as the starting line, landmarks (e.g. the Boston Marathon's notorious Heartbreak Hill), and the finish line. Events may also include interactions with other runners such as when runner i overtakes runner j. We assign both textual and graphical symbols to denote each of these events:

- $\texttt{Starts}(a_i)$, shown graphically as ▶.
- $\texttt{PassesLandmark}(a_i, l_k)$, shown graphically as $\blacktriangle_{l_k}$.
- $\texttt{Finishes}(a_i)$, shown graphically as ■.
- $\texttt{Overtakes}(a_i, a_j)$, shown graphically as .
- $\texttt{WinsRace}(a_i)$, shown graphically as 🏆.

Example 5: Continuing from Example 2, in the kindergarten recital scenario, we might consider events that model important happenings within the performance, both planned ($\texttt{SingsSolo}(a_i)$) and unplanned ($\texttt{MakesSillyFace}(a_i)$).

Example 6: Continuing from Example 3, the events of interest in the marketplace scenario might be visits to individual vendors ($\texttt{Visits}(a_i, v_k)$) and passing interactions with other actors ($\texttt{SpeaksTo}(a_i, a_j)$).

Note that, although events occur at particular times and places, and even though those details are quite important for the observers (see below) to be in place to capture events, at a fundamental level our model for events is symbolic, rather than spatial. That is, we consider all events of the form, say $\texttt{Overtakes}(a_1, a_2)$, to be essentially equivalent, regardless of when and where those events occur. The essential element is the *sequence of events*, rather than where those events occur in time and space.

C. Observers

To capture the events, a collection of *robots* $R = r_1, \ldots, r_m$, each bearing at least one camera, move autonomously through an environment, either in a distributed fashion or under the direction of a centralized controller. We use the term 'robot' very broadly, to include stationary cameras and cameras mounted upon pan-tilt platforms, along with traditional wheeled or aerial robots.

At time t, each robot r_i has a *field of view* $F(r_i, t) \subset W$, indicating the portion of the environment visible to robot r_i at time t. If an event e occurs at time t, within $F(r_i, t)$, then we say that r_i has *captured* e.

Though the robots can move as time passes, their movements will, in general, be constrained by velocity limits or by the robots' own kinematics. Such movement constraints are well-studied for mobile robots, and the details are orthogonal to our interests here. Thus, in this paper we abstract away the physical details of robot motion, and consider only the *reachable sets*, denoted $R(r_i, t_1, t_2) \subseteq W$, and indicating the set of possible states reachable by r_i at time t_2, starting from its actual state at time t_1.

D. Story Specifications

We can now finally turn our attention to the question of how observed events combine to form narratives. For our purposes, a *story* is a sequence of observed events, satisfying some structure constraints. A vital preliminary question is to determine how to specify these constraints. We propose, for this paper, to do so by borrowing some ideas from the theory of formal languages.

Specifically, we treat the space of possible events as an alphabet and construct an automaton to describe the sort of narratives that the system should endeavor to capture. (Events either occur or do not, with their specific time of occurrence being unimportant). But, given only some specification of the quintessential story, it remains quite possible that the observers will be unable to capture any fitting story. This could occur because certain important events occurred outside the field of view of all of the observers, or simply because those events did not, in fact, occur. Accordingly we consider a specification as inducing a preference over stories, i.e., across all finite sequences of events. Rather than the strict in/out criterion usual in classical formal languages — we thus employ a weighting to associate a scalar value to sequences drawn from the alphabet. For simplicity we confine our attention to ideas based on regular languages as they are readily represented as finite automata. This also makes it easy to add a scalar weighting, with a semantics that is easily described in what follows (though, only informally).

We provide a finite set of states and also two distinguished subsets: a non-empty set of starting states and another of

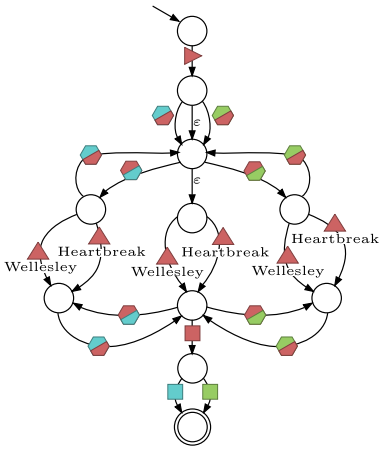

Fig. 2. Pictorial representation of a specification for an interesting race where Alice vies with Bob and/or Candice, and is recorded passing a landmark. Weights have been omitted to minimize visual clutter: symbols all have weight -1, except ε edges are 0 and implicit self-loops are 1. See video at [30].

terminating ones. States are connected with directed edges, each bearing an event and a weight (from $\mathbb{R}$). A sequence of events is evaluated by tracing events in the natural way, from some start state through the automaton, traversing edge by edge, via edges carrying labels with matching events. As the edges are traversed, the associated weights are summed along the way. And, as in the conventional arrangement, special edges marked with ε can be taken for 'free' without consuming an event. An important deviation from the standard model is that, when tracing a sequence and no outgoing edge is found to match, the automaton stays in the same state (along with some fixed, constant weight). These implicit self-loops improve clarity of the specification, enabling one to focus mainly on describing events that are part of an ideal story, and helps ensure that the process is painless even if there are many different events. Note that there may be multiple valid paths that can be traced on the automaton (e.g., from several starting states, or multiple matching edges, or different choices of where to take an implicit self-loop). Here, the path with minimum weight is considered. We term this quantity the *total traversal weight*.

Example 7: (Building on Examples 1 and 4.) This weekend, Alice will be running a race with Bob and Candice, two office frenemies. She's not sure about the relative athleticism of either of them, but she desires a good video nevertheless. A specification for a highlight clip to give her good bragging rights around the office water cooler might be the finite automaton in Figure 2. The specification asks to see a duel between Alice and either Bob or Candice, whomever is the evenly athletic runner and hence her nemesis on the day. Ideally, the video would include a scenic shot with Alice running past a recognizable landmark: either cresting the famous Heartbreak Hill, or Alice's alma mater. This shot might be before or after, or even between, the prolonged duel that, though dramatic, ultimately has Alice victorious.

E. Orchestration and coordination of observers

Some subset of the observers may be capable of modifying their recording activities on the basis of direction provided to them. The idea is that, for example, information might be provided to individual mobile cameras, giving them instructions to attempt to record particular events.

The preceding are all the necessary elements to state the narrative capture problem: Given a narrative specification automaton L over an event space E, direct the observers to capture a sequence of events $e_1 \cdots e_k$ that minimizes the traversal weight of $e_1 \cdots e_k$ on L.

IV. Approach

We describe our current approach, which addresses the basic narrative capture problem within the context of a simulated race: it moves robotic cameras to capture events that tell the stories of what happens to actors. We provide some details of the system we have implemented, beginning with an overview of the system architecture, then provide a few details on the planning and coordination elements. Then we present some demonstrative "video" clips.

A. Planning system overview

A schematic of the overall system architecture appears in Fig. 3. The green block, the planner, is the core component, being the locus of central decision-making. It examines an aggregation of the events recorded thus far, determines which feasible events would improve the current standing, and communicates with one or more observers, each operating in the environment, by sending them high-level commands. To demonstrate *coordinated* event observation, we assume that there are multiple observers. In our treatment, the observers are essentially robotic cameras, each being basically autonomous: directives are sent to cameras, but are at a high-level ("It appears that $\texttt{Overtakes}(Bob, Jack)$ might occur at mile marker 4 in 90 seconds time."). The observers know of their own details (constraints: field of view, motion limits; state information: current position and velocity), and have a local controller that attempts to capture the requested events.

The planner sends directives by combining three aspects:

(1) It examines an aggregation of the events that have been recorded thus far, in the schematic these are the

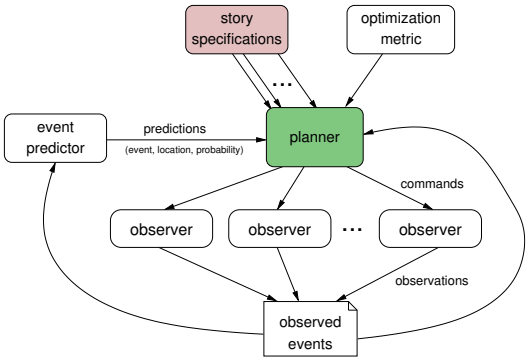

Fig. 3. System architecture. The planner, in green, processes observations and predictions of future events to direct the observers where to go. The desired story is achieved via an optimization process to maximize the match to provided story specification (red block). The final video post production is performed offline.

observed events, or, in cinematographer's parlance, the can. They are stored simply as a sequence, ordered by time. (Only the symbol representing the successful capture of the event being captured is needed, not the observer's specific observation.)

(2) The planner consults a set of story specifications (shown in red), along with an optimization metric, provided at the start. We permit multiple specifications because there may be many stories (e.g., while Alice has an idea for a highlight video, Bob and Candice could conceivably have their own ideas as well). The optimization metric provides a way to prioritize efforts among the collection: is the system aiming to generate the most compelling story, a "Pulitzer-prize" type reward model (metric: `max`), or to produce multiple good stories on average (`ave`), or to maximize the number of narratives with scores above a threshold, to maximize the number of vanity videos produced (`min`)?

(3) The planner receives messages from a module called the event predictor. The predictor uses the observed events and a model of actor state, and space and time, to forecast.

The planner uses these pieces of information to ask which potential future sequence of events would improve the story that can be constructed. It then determines which observers should be instructed to attempt to collect these events, and sends our those directives.

The final part, which we call post production, is not shown in the schematic because it operates after all the events have been captured. The specific events that comprise a story must be stitched together to produce an output. At this point, other additional stylistic aspects can come into play: if multiple observers captured an event from different vantage points, the best one should be selected; when two sequences are brought together an appropriate transition must be made between them, whether it be a dissolve, fade, or cut.

B. Planner: Some details

We elaborate further on some specific details of the planner's operation.

The planner takes the sequence of events that been recorded thus far (or the empty sequence at the inception) and traces them on the specification automata to compute the total traversal weight. Next, it computes the marginal improvement that would result if some of the events provided by the event predictor, for a short duration in the future, were captured. It calculates this by tracing these predicted events forward from the state(s) reached by the already recorded events. From the states reached by those events, the least weight cost to some terminating state is found. The weights in the future portion of the trace are multiplied by a 'future' discount factor (we use 0.5 in our implementation). By comparing different possible sequences, we obtain an estimate of how much the predicted events, if captured, would contribute to a story when it is finally completed. (For maximal simplicity, in our implementation, the planner is greedy and so the evaluated sequences of predictions all consist of only one prediction.)

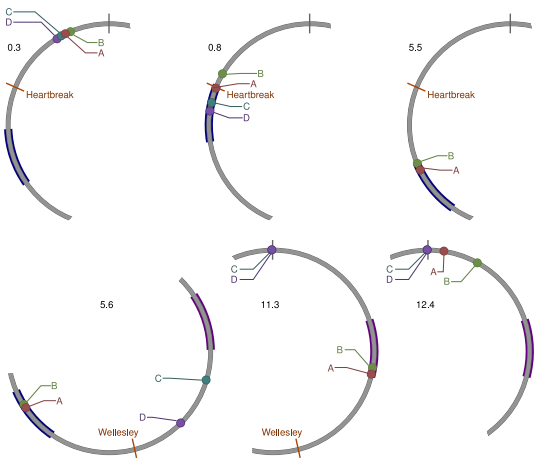

Fig. 4. Overhead view of simulated three lap race showing four runners, and three cameras, only of two which are shown, re-position themselves to capture the footage used to produce Figure 5. Greater (temporal) detail is included for the overtaking events which shows how the predictors have provided information used to assign cameras to those events and the cameras move to capture that footage.

We impose a threshold on the improvement values to whittle down the list of predicted events to the subset that are most valuable. For each of these, an estimate of the cost of the cameras is estimated. For the mobile cameras, this is expressed in terms of distance from the event; for immobile ones, the cost is free for items already within view, and infinite otherwise. Next an optimal assignment problem is formed and solved using Munkres's algorithm [31]. We have found that it suffices to have the planner repeat this at each time step, simply re-assigning cameras to event locations as predictions change.

The predictor in our implementation keeps an estimated position and velocity for each runner, and it uses this to predict their positions at future points in time. It uses the position and velocity of each runner from the last time they were recorded as part of any event, smoothed with an exponential moving average filter. The predictor can use this to give estimates of when a given runner will finish the race (based on knowledge of the position of the finish line), pass a particular landmark, overtake or be overtaken by another.

C. Simulation

The system described was implemented in Python, along with a simulation of a multi-lap race on a circular track. Snapshots showing the overhead view of the simulation appear in Figure 4. Those figures show a race with four runners, being the scenario in Example 7, but with a fourth runner unknown to the other three. In this scenario, there are three cameras: a fixed camera at the starting line and two moving cameras. Since the track is single dimensional, each "camera" is a 1D vector representing the camera's limited field of view. Runners appear as pixels, colored according to the runner. Landmarks appear as pixels of a different color. A "video" is a 2D array with time as the

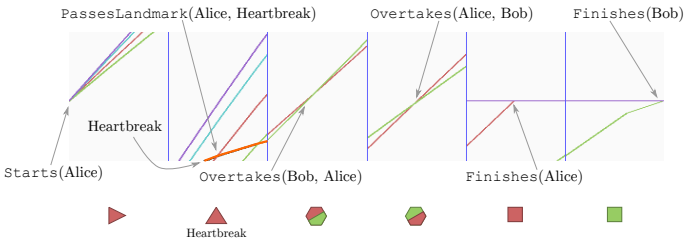

Fig. 5. Output "video" from a four-runner race using specification in Figure 2. Legend: the vertical blue lines are where footage cuts away from one camera to another; the orange lines show the landmark within the frame; Alice is the red runner, Bob is the green one. See video at [30].

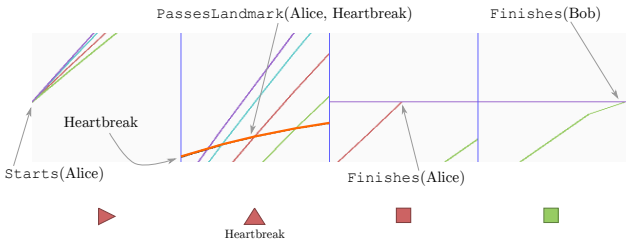

Fig. 6. A second output from an identical race to Figure 5, but in this case the system does not have precise pose estimates for the actors (e.g., from tracking chips within runners' shoes); consequently their are too few observations to make predictions of the overtaking events with any precision.

horizontal axis and individual camera frames as the vertical axis. Using the specification in Figure 2 for Alice, we produced two highlight videos for her. (We only considered a single specification, so the choice of optimization metric is irrelevant.) They are shown in Figures 5 and 6.

The first, Figure 5 captures the drama of Alice's duel with Bob, including her final victory. For cameras to capture the vital overtaking events, the predictor needed to have recent and fairly reliable information about the positions and speeds of the runners. We achieve this by simulating a race tracker, via devices such as the Arion running shoe (See `getarion.com`), which was easily added by introducing a different sort of observer. Without this detailed information, overtaking predictions are scant and the system produces a video like that in Figure 6. Though less involved, and with a more limited dramatic arc, it is still clearly a customized video that adheres to the structure specified in Figure 2.

D. Post production (rendering the video)

The output video is constructed by first identifying events to include — which involves tracing the path that gives the minimal total traversal weight for the events in the can. Currently, the video is then composed by switching between cameras at midpoints between events. If the video is too long, we reduce the time before and after the selected events. Though we do not yet do it, if the video were too short, we could use slow motion instant replays to highlight key events.

V. Hardware Proof of Concept

We conducted a basic proof-of-concept demonstration of the techniques from Section IV. We held a race using four Anki Overdrive robots on a race course loop. An overhead camera operating at 30 FPS captured the entire race, and a computer vision script detected the position of each racer

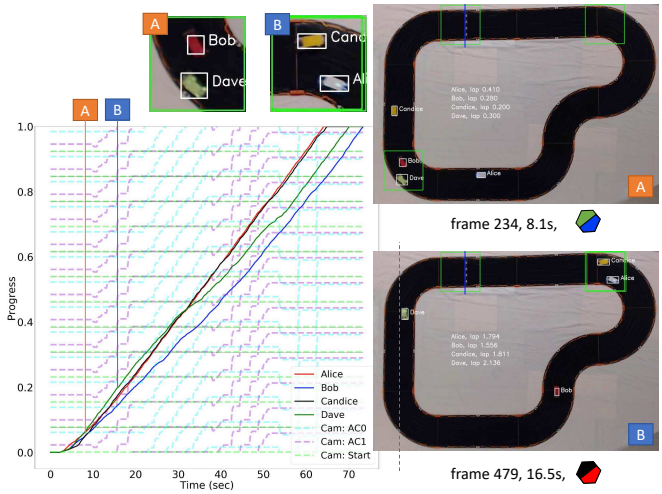

Fig. 7. (Right) Two frames from an overhead camera filming the robot race. Frames are annotated with racer position and progress. Green outlines show the three virtual cameras. Plot shows the progress of all racers and the position of the virtual cameras as a function of time during the 13 laps.

(Alice, Bob, Candice, Dave) and computed their progress during a 13-lap race. This information was used as GPS data for the event predictor and planner. Our image processing script also detected events.

The robot race course has a centerline distance of 3527 pixels (480.7 cm long). The race cars are each 8.2 cm long and have a maximum speed of 4.2 cm/s (30.5 pixels/second). Each race car was equipped with a colored fiducial to simplify tracking based on color swatches. We emulate camera movement by using a moving region-of-interest (ROI) to represent each virtual camera. Our routine used three virtual cameras, each with a ROI 200×200 pixels. The cameras had a maximum speed that was 70% the speed of the fastest racer, and were allowed to overlap and pass each other. Screenshots and a summary of the race are in Fig. 7.

VI. Conclusion

This paper presented the new problem of online planning for a set of cameras to gather sensor data to fit some story structure. This narrative structure was induced via an objective function specified as a weighted automaton. We presented an example application in simulation and hardware, and showed that though the precise story that can be output depends on the information available to the planner, the result still conforms to a reasonable structure as expressed in the specification.

Designing a good specification can be challenging, however, and future work should provide tools for good storytelling specifications. If [1] is correct, and there are a limited set of legitimate narratives, this task may be simplified substantially. Though we have provided a design capable of producing multiple stories simultaneously, the preliminary simulation in this work only demonstrated optimization of a single narrative. Understanding how to simultaneously collect footage for a variety of outcomes remains an interesting question addressed only partially in [14], [32].

REFERENCES

[1] Vladimir Yakovlevich Propp. *Morphology of the Folktale*, volume 9. University of Texas Press, 1968.

[2] Alan Dundes. On computers and folk tales. *Western Folklore*, 24(3):185–189, 1965.

[3] Pablo Gervás. Propp's Morphology of the Folk Tale as a Grammar for Generation. In Mark A. Finlayson, Bernhard Fisseni, Benedikt Löwe, and Jan Christoph Meister, editors, *2013 Workshop on Computational Models of Narrative*, volume 32 of *OpenAccess Series in Informatics (OASIcs)*, pages 106–122, Dagstuhl, Germany, 2013. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

[4] Yong Jae Lee, Joydeep Ghosh, and Kristen Grauman. Discovering important people and objects for egocentric video summarization. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2012.

[5] Chong-Wah Ngo, Yu-Fei Ma, and Hong-Jiang Zhang. Automatic video summarization by graph modeling. In *Proc. IEEE International Conference on Computer Vision*, 2003.

[6] Richang Hong, Jinhui Tang, Hung-Khoon Tan, Chong-Wah Ngo, Shuicheng Yan, and Tat-Seng Chua. Beyond search: Event-driven summarization for web videos. *ACM Transactions on Multimedia Computing, Communications, and Applications*, 7(4):35, 2011.

[7] Danila Potapov, Matthijs Douze, Zaid Harchaoui, and Cordelia Schmid. Category-specific video summarization. In *Proc. European conference on computer vision*, pages 540–555. Springer, 2014.

[8] Boqing Gong, Wei-Lun Chao, Kristen Grauman, and Fei Sha. Diverse sequential subset selection for supervised video summarization. In *Advances in Neural Information Processing Systems*, 2014.

[9] Michael Gygli, Helmut Grabner, and Luc Van Gool. Video summarization by learning submodular mixtures of objectives. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[10] Peng Chang, Mei Han, and Yihong Gong. Extract highlights from baseball game video with hidden markov models. In *Proc. International Conference on Image Processing*, 2002.

[11] Maheshkumar H Kolekar and Somnath Sengupta. Event-importance based customized and automatic cricket highlight generation. In *Multimedia and Expo, 2006 IEEE International Conference on*, pages 1617–1620. IEEE, 2006.

[12] Zheng Lu and Kristen Grauman. Story-driven summarization for egocentric video. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2013.

[13] David Cottrell. System and method for automated creation of video game highlights, August 20 2013. US Patent 8,515,253.

[14] Yogesh Girdhar and Gregory Dudek. Efficient on-line data summarization using extremum summaries. In *Proc. IEEE International Conference on Robotics and Automation*, 2012.

[15] Yogesh Girdhar, Philippe Giguere, and Gregory Dudek. Autonomous adaptive exploration using realtime online spatiotemporal topic modeling. *International Journal of Robotics Research*, 33(4):645–657, 2014.

[16] Misha Rabinovich and Y Girdhar. Gaining insight into films via topic modeling & visualization. *Parsons Journal of Information Mapping*, 7(1):3–5, 2015.

[17] Jingjin Yu and Steven M LaValle. Cyber detectives: Determining when robots or people misbehave. In *Algorithmic Foundations of Robotics IX*, pages 391–407. Springer, 2010.

[18] Jingjin Yu and Steven M LaValle. Story validation and approximate path inference with a sparse network of heterogeneous sensors. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 4980–4985. IEEE, 2011.

[19] Hannaneh Hajishirzi, Julia Hockenmaier, Erik T. Mueller, and Eyal Amir. Reasoning in Robocup Soccer Narratives. In *Proceedings of 27th Conference on Uncertainty in Artificial Intelligence (UAI'11)*, 2011.

[20] Stephanie Rosenthal, Sai P. Selvaraj, and Manuela Veloso. Verbalization: Narration of autonomous robot experience. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence (IJCAI'16)*, pages 862–868, 2016.

[21] Mark O. Riedl and R. Michael Young. Narrative Planning: Balancing Plot and Character. *Journal of Artificial Intelligence Research*, 39:217–268, 2010.

[22] Nicholas D. Allen, John R. Templon, Patrick Summerhays McNally, Larry Birnbaum, and Kristian J. Hammond. Statsmonkey: A data-driven sports narrative writer. In *Computational Models of Narrative, Papers from the 2010 AAAI Fall Symposium, Arlington, Virginia, USA, November 11-13, 2010*, 2010.

[23] Arnav Jhala and R. Michael Young. Cinematic Visual Discourse: Representation, Generation, and Evaluation. *IEEE Transactions on Computational Intelligence and AI in Games*, 2(2), June 2010.

[24] Sherol Chen, Adam M. Smith, Arnav Jhala, Noah Wardrip-Fruin, and Michael Mateas. RoleModel: towards a formal model of dramatic roles for story generation. In *INT3'10: Proceedings of the Intelligent Narrative Technologies III Workshop*, pages 1–8, New York, NY, USA, 2010.

[25] N. Szilas, M. Axelrad, and U. M. Richle. Propositions for Innovative Forms of Digital Interactive Storytelling Based on Narrative Theories and Practices. *Transactions on Edutainment*, VII:161–179, 2012.

[26] Hong Yu and Mark O. Riedl. Personalized Interactive Narratives via Sequential Recommendation of Plot Points. *IEEE Transactions on Computational Intelligence and Artificial Intelligence in Games*, 6(2), 2014.

[27] A. Amos-Binks, C. Potts, and R. M. Young. Planning Graphs for Efficient Generation of Desirable Narrative Trajectories. Working Notes of the AIIDE Workshop on Intelligent Narrative Technologies, 2017.

[28] J. Robertson and R. M. Young. Narrative Mediation as Probabilistic Planning. Working Notes of the AIIDE Workshop on Intelligent Narrative Technologies, 2017.

[29] C. Barot, M. Branon, R. E. Cardona-Rivera, M. Eger, M. Glatz, N. Green, J. Mattice, C. Potts, J. Robertson, M. Shukonobe, L. Tateosian, B. R. Thorne, and R. M. Young. Bardic: Generating Multimedia Narrative Reports for Game Logs. Working Notes of the AIIDE Workshop on Intelligent Narrative Technologies, 2017.

[30] Dylan A. Shell Jason M. O'Kane Li Huang, Aaron T. Becker. Make robots tell a story: Planning coordinated event observation for structured narratives.

[31] J. Munkres. Algorithms for the assignment and transportation problems. *Journal of the Soc. for Industrial and Applied Math.*, 5(1):32–38, March 1957.

[32] Florian Shkurti, Nikhil Kakodkar, and Gregory Dudek. Model-based probabilistic pursuit via inverse reinforcement learning. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7804–7811. IEEE, 2018.