Robot Magic: A Robust Interactive Humanoid Entertainment Robot

Kyle J. Morris[1], Vladyslav Samonin[1], John Anderson[1], Meng Cheng Lau[1], and Jacky Baltes[2]

[1] Autonomous Agents Laboratory
Department of Computer Science
University of Manitoba
Winnipeg, Manitoba, R3T 2N2, Canada
http://aalab.cs.umanitoba.ca/

[2] Department of Electrical Engineering
National Taiwan Normal University
Taipei, Taiwan

Abstract. In recent years, there have been a number of popular robotics competitions whose intent is to advance the state of research by comparing embodied entries against one another in real time. The IEEE Humanoid application challenge is intended to broaden these by allowing more open-ended entries, with a general theme within which entrants are challenged to create the most effective application involving a humanoid robot. This year's theme was Robot Magic, and this paper describes our first-place winning entry in the 2017 competition, running on a ROBOTIS OP2 humanoid robot. We describe the overall agent design and contributions to perception, learning, control, and representation, which together support a robust live robot magic performance.

1 Introduction

There are many ways of advancing robot technology and artificial intelligence, from studying component elements (mechatronics, kinematics, vision, learning) to building fully-functional agents for particular domains. The history of artificial intelligence research has generally shown the latter to be especially important, since studying components in isolation removes the effects of interaction with other components, and excludes factors that only embodiment brings. This is one of the main motivations for the evaluation of intelligent robotics in competition settings [1], and such competitions have become both an important part of evaluating robotics research and as well as a driver for public interest in robotics and artificial intelligence.

A broad range of robotics competitions exist today, differing by intended audience and goal. The most well known of these are robot soccer competitions such as RoboCup, as well as more specialized competitions (e.g. RoboCup@Home for service robotics) and those emphasizing a broad range of skills (e.g. the FIRA

HuroCup, a multi-event robot athletic competition intended to encourage breath in humanoid performance [2]).

The IEEE Humanoid Application Challenge is a humanoid competition designed to be more open-ended, and incorporate potential avenues of research not well represented in other competitions, from dexterity and complex motion planning to believability and human-robot interaction. In the last two years, this event has involved the area of entertainment robots as a common ground between entries, with a theme of robot magic - where a humanoid robot can take on any role in a magic show.

Magic is well-known form of entertainment and deception, with tricks for amusement and as a illusion of power possessed by a magician having been recorded from the ancient world through to modern times [3]. It is a cross-cultural activity, and in that sense is something that is accessible to a broad audience, similar to soccer. Magic employs a range of techniques, from slight of hand to physical devices to create these illusions. A key part of any magic act is also distracting or changing the focus of an audience to disguise the mechanisms behind a given trick, a form of disinformation provided through the principles of stagecraft [4].

All of this provides much opportunity for improving work in robotics and artificial intelligence. A plausible robot magic act also has strong connections to other real-world domains: perceiving an error-prone world, goal directed reasoning, adaptation in the face of failure, and successful interaction - all performed in real time under challenging conditions.

We have been successful in our efforts in the Humanoid Application Challenge over a long period of time, having taken first or second place in each year this competition has been run. We most recently placed first in the 2017 Robot Magic Competition, and this paper describes the agent design of this winning entry and its implementation on a ROBOTIS OP2 Humanoid Robot.

2 Related Work

There are a broad range of applications of artificial intelligence that can be considered entertainment-related. There is work on producing art and composing music (e.g. [5]), but these are distinct from live interactive works in that it is the artifact produced by the system - the completed music or art - that is intended to be entertaining to humans, rather than the process (though there are real-time efforts as well, such as Google's *AI Duet* and *Quick, Draw!* [6]). There are also intelligent systems built for playing many games against humans, such as traditional board games such as checkers [7], chess, and a broad range of video games.

While these may be considered live performances, they are distinct from working with robots in a real time setting, since interaction is generally virtual, through a computational user interface rather than a face to face interaction. The challenges are much greater when working with physical robots - sensors are erroneous, items being moved can be dropped, robots trip and fall, batteries

run out of power, and all manner of real-world interaction can occur that does not need to be considered in virtual environments. Any physical interaction requires significant systems to be engineered (e.g. vision that is robust to lighting changes).

Because of this, instances of viable physical entertainment robots are less numerous than their software counterparts, and working with humanoid robots in particular introduces many challenges. Balancing is much more difficult than using wheeled robots, for example, not just because of a bipedal walk but because of the complex poses the body can achieve. Beyond balancing, the more complex motion model of a humanoid (and sensors mounted on head that move independently) greatly complicates fundamental robot applications such as simultaneous localization and mapping (SLAM) [8, 9]. Such applications are necessary, for example, for a humanoid robot to position itself on a stage during a performance, or manage objects within its space. For that reason, human-like robots intended for interaction with humans (including entertainment) are frequently only partially humanoid in form, with a more rigid body structure and substituting a wheeled platform for legs (e.g. Pepper [10]).

There are some entertainment-related examples of humanoid robots that are not simply toys. For example, Kuroki [11] presented a bipedal robot intended for entertainment purposes, but the core of the work involved complex motion planning so that the robot could perform motions interacting with music, as opposed to interacting with humans directly.

In terms of work directly with magic, there are few directly comparable published works. Koretake [12] presented a robot intended to be used for card magic, but this work focused only on card manipulation through a device without anthropomorphic hands, and with no humanoid body. Even within the context of card magic, the work involved only manipulating cards directly (e.g. second card dealing), as opposed to any means of timing or interaction in an overall magic performance, nor anything beyond card magic. There has been growing discussion of the need for timing and human-robot interaction for effective live performance [13, 14], but this discussion has been largely theoretical.

The previous (2016) IEEE Humanoid Application Challenge was also themed on robot magic, and our entry was a humanoid that could autonomously act as an assistant to a human magician [15]. The primary contribution in that work was on machine learning to recognize cards during real time interaction, as opposed to prior work where cards were expected to be reviewed from a fixed distance and angle and required latency that would be unacceptable in a live performance, and an engine for live scripted performance. Our work here furthers this significantly: making the robot central as the magician, focusing on a representation that allows multiple magic tricks of different types to be performed, and improving interaction mechanisms and robustness in the performance. The result is an approach that, while demonstrated as a live magic act, can function broadly in domains that require scripted entertainment and interaction with an audience.

3 Agent Design

In this section we present an architecture that allows for developing robust and extendable live entertainment - while we focus on magic in this particular competition, there is nothing restricting this approach to that one application.

Our architecture is implemented on a ROBOTIS OP2 with 22 degrees of freedom (DOF) and an on-board Atom-based PC. Sensing is provided by an accelerometer and gyroscope for balancing, and a camera, speaker, and microphone for interaction. The robot in performance is depicted in Fig. 1.

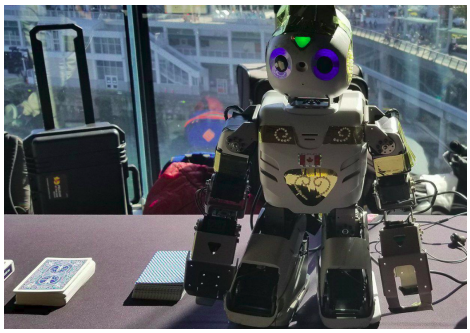

Fig. 1. The robot at a live show at IROS 2017, Vancouver.

Our agent architecture (Fig. 2) is centered around live performance, and allows for exploring the notion of a fully embodied agent that perceives, plans, and acts in an entertainment environment. The entertainment, in this case a magic performance, is developed as a series of behaviors structured in a tree. behaviors can be linked to one another, and are internally structured as finite state machines (FSMs). These collectively represent a show consisting of a series of episodes (magic tricks). Each FSM state may contain various submodules which utilize perception, learning, and robot control. The state transitions may be adjusted dynamically according to high level goals, allowing for reaction to unexpected conditions. Through these, the robot is taken through all the necessary motions and interactions with the audience to present the show. The following subsections describe the major components in this approach.

3.1 Perception Module: Supporting Recognition and Interaction

The perception module receives incoming sensory information and acts as an interface to the planning and learning modules. We extended previous work [15] to create an API for playing card classification, speech recognition, and raw capturing of video and audio input. Perception in terms of the particular objects

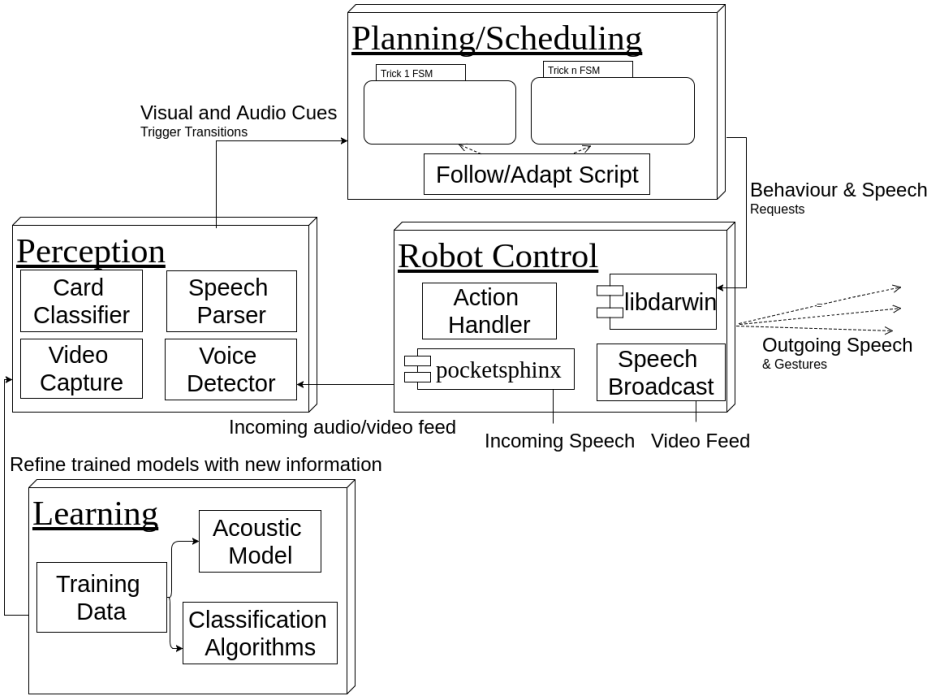

Fig. 2. High Level Agent Architecture

of interest and methods of interaction is domain specific, and here we cover the work done for a magic performance.

Much of the specific image processing in our work is recognizing cards in real time, so that they can be chosen or shown to an audience. Raw frames are received and stored in an OpenCV Mat after applying gray scaling, blur, and binary thresholding. Front-of-Card recognition is acquired by detecting contours on the input image, and doing a polygon approximation to gather estimated card corners. After applying feature checks and performing an affine transformation, the card suit and rank regions of interest are extracted and sent to the learning module for classification. Working with the rear images of cards is also important for many games, such as when dealing cards. Back-of-card detection undergoes the same preprocessing phase. However, blob detection is instead used to recognize the dotted pattern of the cards back. Each "dot" represents a blob on the back of the card. Among these blobs, the centroid is calculated, and if the cluster of blobs surpasses the minimum density threshold, we deem the card as found.

For speech, raw audio is sampled at 16 mhz and hypothesis strings are then created using the PocketSphinx engine. After a hypothesis is formed, it is mined for keywords in the learning module, and encoded information (detected ob-

jects/keywords in speech) that provoke state transitions in the performance. Latency in this component is reduced by performing a window-based sampling of the input speech, instead of waiting for a given speech instance to terminate. In the traditional approach, speech is sampled until there is a lack of speech input. At this point a hypothesis string will be formed. The challenge with this is that in the presence of a loud environment (e.g. audience noise), the speech sample may continue indefinitely as there is no well defined gap in audio input. Waiting for the entire utterance also introduces a significant lag in response in a live performance. We adapt this by forcing a sample window to cut the speech apart in fixed intervals, where mining for keywords may be performed on multiple windows separately. This guarantees key information will be detected within a given window, and promotes timely response in a live performance.

3.2 Learning Module: Adapting to the Environment

The learning module extracts features and performance-critical information from preprocessed perceptual input. Offline learning utilizes vision and speech samples taken from the perception module to train on an ensemble of machine learning models. Training data for playing cards is created using the robot's onboard camera and stored in CSV format. In our system we use CMUSphinx to create a trained acoustic model for the assistant's voice, that will then be used by PocketSphinx for speech recognition during the performance (for the assistant as well as the audience members). Training for playing card recognition uses an ensemble of K-Nearest Neighbours[16] and Logistic Regression. The learning module furthermore acts as a working-memory during the performance by storing run-time information such as card values and audience responses that are required for provoking transitions between behaviors.

3.3 Control Module: The Hardware-Software Interface

The control component provides drivers and lower level primitives for interacting with the robotic hardware. This allows for interfacing the perceptual module with various devices and acts as a middleware in our system to address robot components abstractly. This was necessary as major robotic development tools such as ROS do not yet support the ROBOTIS OP2. We developed an Action Handler built on-top of the libdarwin Framework (the basic action framework built into the ROBOTIS OP2) that allows for simultaneous motion and speech. Audio input is recorded through a NESSIE Adaptive USB Condenser Microphone and the PulseAudio sound server. OpenCV is used to read in video feed from the ROBOTIS OP2's Logitech camera. Sensory input is then passed to the perceptual module where it is preprocessed and encoded for the learning module.

3.4 Planning Module: Magic Tricks as Finite State Machines

The Planning/Scheduling module causes the robot to achieve goal directed behavior and adapts this as necessary during the show. Like any performance, this

involves having a representation of the performance and adapting this as the performance unfolds.

Most sophisticated applications intended to run on a robot must consider high level goal-directed reasoning (e.g. planning to pick up an object or move to a location). At the same time, the actions manipulated by a planner are not usually directly executable on a robot as a single instruction (for example, in a humanoid a basic action such as moving forward a discrete amount requires adjusting many servos with careful timing and integrating this with sensors for balance). In our approach, the lowest level of this is handled by the Control Module's Action Handler. Above this we employ a behavior-based approach [17], where behaviors are interconnected and form a context and influence related or lower-level behaviors. Behaviors are constructed as finite state machines, which are in turn connected into larger behavior tree structures, analogous to their use in other types of dynamic activities, such as robot soccer [18].

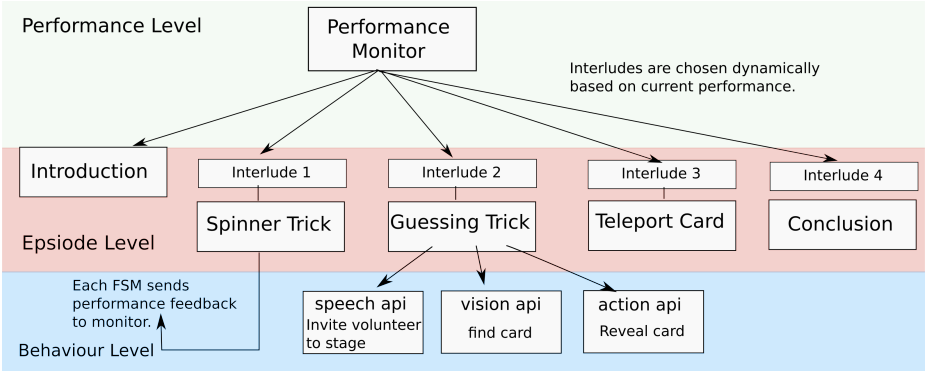

Fig. 3. Structure of a performance

These trees form a three-level hierarchy in our approach, shown in Fig. 3. At the *Performance* level, the objective is to complete a performance within a given time constraint, and maintain audience involvement. A performance consists of an introduction, a number episodes (in our domain, magic tricks or illusions), and a conclusion. This top level acts as a supervisory monitor that overviews the progress in the live show. Below the Performance level is the *Episode* level. Here we focus on an individual portion of the performance, and the objective in our demonstration domain is to successfully complete a single magic trick. Below the episode level, the *Behavior* level interfaces with the control module for performing actions, and interfaces with the perception module to gather information at run time. At a high level, a magic performance can be seen as a finite state machine with transitions between all of the tricks, and connecting interludes in the show. The performance layer adds robustness by supplying a context to the behaviors below it. For example, it can allow transitions between

states to change dynamically given the progress in a performance. These may be high-level modifications such as skipping a magic trick in the event that time is running short, or low-level adjustments such as changing a dialog to account for the transition between two potentially unrelated tricks. For example, if trick A was related to trick B, but B is cancelled, a new interlude from A to C is required). This handles a range of dynamic alterations, similar to altering a planned walking path when a new obstacle is discovered.

Performance episodes are domain dependent, and require representing the trick (or other element of performance in a different domain) in terms of one or more behaviours (finite state machines). A verbal description of a magic trick might be:

> *The robot selects a volunteer from the audience and picks up the magic deck. The volunteer picks a card from magic deck. The robot asks the volunteer to show the playing card to the audience. The robot waits for the assistant to provide a cue. The robot will determine card suit and type from visual and auditory cues. The robot will guess the card suit and type, and confirm with the volunteer by revealing the card.*

Fig. 4 shows the structuring of this as a finite state machine. This example requires no cycles and only one branch, but these can be arbitrarily complex. The individual components may be links to additional behaviors, or direct activities in terms of robotic instructions or perception/interaction requests. Currently, this representation must be constructed by the developer of the performance, but we are working on adopting automatic XML to behavior tree generation, which has previously been used in other domains [19].

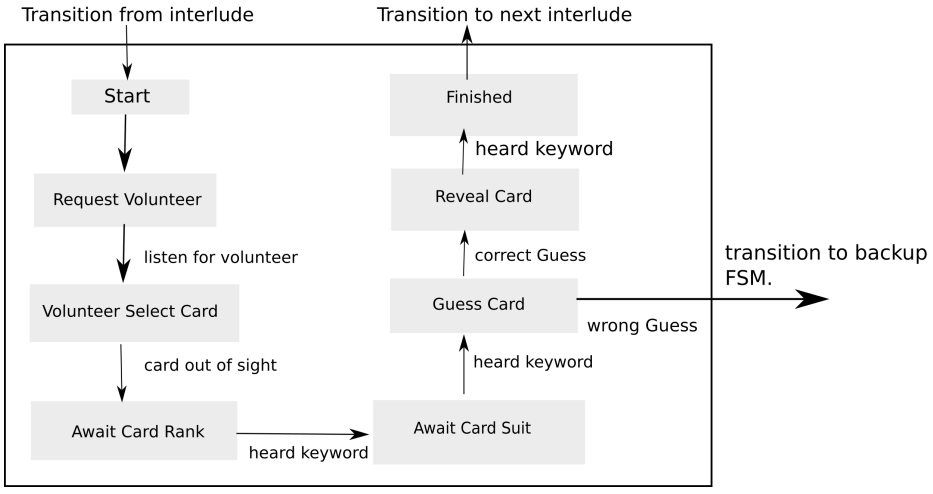

Fig. 4. Example FSM for a magic trick/performance episode.

3.5 Fault Tolerance and Robustness

Like any other robotics domain, a large range of potential problems can occur during a live entertainment performance. The fact that this is carried out in front of an audience, and the unfolding events affect their expectations, makes this even more challenging. Ultimately, no matter what happens, the show must go on. Part of any robust performance involves ensuring the representation of activities is as detailed as possible so that potential problems can be anticipated and planned for. However, there is still always the opportunity for mistakes on behalf of the magician, functionality problems with props, or unexpected behavior from the audience or a volunteer.

In our implementation, we limit the scope of uncertainty in the environment to two types of common mishaps that may occur during a magic show. The first is an error on behalf of a human magician, assistant or volunteer (e.g. script lines are forgotten, or a non-useful response is provided). The second type of error involves hardware or unexpected internal behavior in the robot (e.g. microphone disconnects, battery is low). For the first type of error, the planning component can be used to create a dynamic link to a FSM consisting of backup states, which support recovery to unexpected audience or volunteer responses from a range of tricks. For example, if a volunteer is asked to pick a number within some range, and they select a value outside that range, the robot can react accordingly and return to the trick. These backup states provide additional dialog which acts as a natural bridge from a hiccup in a performance to a reasonable checkpoint.

Some instances of the second type of error can be handled similarly. For example, a backup state is used if lighting is too poor for object recognition, causing the robot to say "My vision isn't working, could someone check the lights?", alerting to the problem and buying time to raise the lights. Beyond these, we also monitor components on the robot from the control module and save checkpoints throughout the show. In the event of an unexpected microphone disconnection, vision crash, or unexpected termination of the main process, the performance may be resumed from the most recent checkpoint by transitioning into a recovery state where the robot will explain that "mistakes happen, we are all humans here" before returning to the checkpoint state using multiple ϵ-transitions (transitions that do not require a specific trigger for each) through the performance FSM. This recovery process can be seen in Fig. 5.

If there is no known recovery possibility, the planning component can still dynamically alter links between behaviors to allow the show to proceed based on knowledge of what is wrong and what facilities tricks require. For example, if the vision system crashes at a given point in the first trick, this may not be an immediate problem if vision is not needed, but would strongly affect future tricks. If this cannot be recovered from, a dynamic transition can be placed by the planning module to skip upcoming tricks requiring vision if the vision module has not been detected as working by that point the performance. This supports maintaining what is possible in the show, while also allowing time for problems on the robot to be fixed.

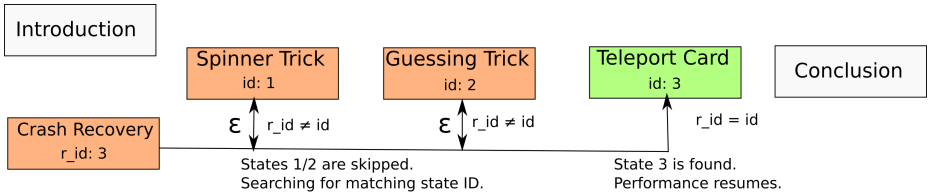

Fig. 5. An outline of the checkpoint recovery system. Upon a run-time crash, we may do multiple ϵ transitions searching for a recent checkpoint.

4 A Full Magic Act

As an example of the breath that this approach can support, we created a magic performance for for the 2017 IEEE Humanoid Application Challenge. Our performance involves four magic tricks, each focused on a different aspect of robotics for entertainment.

Trick 1: Card Spinning: The first trick requires an audience member to choose a playing card, which is shown to the audience but not the robot. This card is placed face down on a fidget spinner toy and provided to the robot, which is tasked with guessing the card's suit and rank. The robot uses back-of-card detection to transition through the trick. The robot determines the suit and rank of the card through visual and vocal cues from the magician (which are apparent to the audience). When the audience member reveals the card, they find that the pattern on the card has been altered, and changed into a swirled pattern to imply that the fidget spinner rotated the cards face. This breaks the ice with the audience by appearing to not perform very good magic (because the robot is told the card) while surprising with a different manipulation.

Trick 2: Guessing Trick: The second trick creates the illusion that the audience actually expected from the first: the robot determines the card rank and suit without appearing to be explicitly told. This is a common human magic trick, and is done by the assistant using coded key words recognized by the robot to convey the particular suit. At this point, the robot fakes stumbling on the performance. The assistant then claims that this mistake shouldn't be happening, as they've been rehearsing this performance since $*MONTH*$, providing a second recognized code that allows a range of 1..12 to be recognized (for cards Ace...Queen). An alternative special keyword is used for the case of a King. The audience does not notice that the dialog provides card information to the robot subtly.

Trick 3: Card Vanishing: This trick utilizes the kinematics of our robot, by using a magic-box prop for making a playing card disappear. The robot must navigate the box in such a way to flip the contents and hide the card.

Trick 4: Finale: The finale builds the stage experience by appearing to come after the show is finished. The audience member from trick 3 is thanked, and the robot prompts the volunteer to select a chocolate bar from the snack table

by way of thanks for their help in the performance. Upon picking a chocolate bar, the volunteer finds their vanished card from trick 3 inside the wrapper.

5 Evaluation

There are many ways of evaluating a work such as this, where a large number of components come together into a functional agent. The individual components can certainly be tested under controlled conditions, either individually or in subsets. In our work, for example, we have tested our vision and learning together. Under dynamic lighting, and with the learning mechanisms described above, we achieve 89% card rank accuracy in dynamic lighting conditions, and 90% suit accuracy. Similarly, there is a noticeable performance increase in reduced speech sampling latency resulting from the window-based approach to sampling speech as opposed to waiting for a speech pause.

This is not as easy to do when looking at the complete agent in its domain: it is difficult to come up with a numerical representation of how good a robot is at magic. Evaluating robots in their intended environment is a strong philosophy of most robotics competitions. In robotic soccer, for example, there are a definite set of rules and a team stands or falls by its performance - presumably the winner of a robotic soccer competition is the strongest team. Magic, however, has no hard and fast rules. The qualities that make a strong performance are in many ways inherently qualitative, and this is similar to many human competitions from sporting events such as gymnastics to entertainment-based performances such as music or drama. In the Humanoid Application Challenge, similar issues have always been an issue in terms of judging the most effective application, even before it began using robot magic as a theme. Like many analogous human contests, dispassionate human judges are used to rank entries in terms of metrics suitable to the domain. For robot magic, entrants are judged based on a live performance and a technical presentation, using metrics such as effectiveness, functionality, technical complexity, and audience involvement, and these are combined from all judges. By this standard the approach described in this paper came out very well, winning first place (a ROBOTIS OP3 robot).

6 Discussion and Future Work

In this paper we have described our approach to robust live performance for a humanoid robot. This involves a number of interacting components for perceiving, learning, and carrying out and altering plans for activity. Central to this is a representation for a performance in terms of behavior structures, and the ability to follow these in spite of problems at run time.

This work has been implemented and demonstrated for a magic performance, but magic is not dissimilar to many other applications, such as comedy or drama. This approach can be used for any application relying on a scripted performance by a humanoid that must deal with technical and interaction-based problems

during performances. For example, actors can miss cues, and heckling can interrupt a comedy act, but both of these can be anticipated to some degree.

There is much potential for future work in terms of better learning and interaction mechanisms, improvements in motion control, and the dynamic construction of behaviours to simplify the description of performances. We also intend to explore dynamic performance construction, choosing elements from a known library and altering choices based on what pleases the audience most.

References

1. Baltes, J., Anderson, J.: Advancing artificial intelligence through minimalist humanoid robotics. In Liu, D., Wang, L., Tan, K.C., eds.: Design and Control of Intelligent Robotic Systems. Springer-Verlag, Heidelberg (2009) 355–376
2. Baltes, J., Tu, K.Y., Sadeghnejad, S., Anderson, J.: Hurocup: competition for multi-event humanoid robot athletes. Knowledge Engineering Review (2016) 1–14
3. Randi, J.: Conjuring. St. Martin's Press (1993)
4. Tufte, E., Swiss, J.I.: Explaining magic: Pictorial instructions and disinformation design. In Tufte, E., ed.: Visual Explanations. Graphics Press, Cheshire, CT (1997)
5. Liang, F., Gotham, M., Johnson, M., Shotton, J.: Automatic stylistic composition of bach chorales with deep lstm. In: Proceedings of the 18th Int. Society for Music Information Retrieval Conference (ISMIR-17), Suzhou, China (October 2017)
6. Google: Experiments with Google. `https://experiments.withgoogle.com/ai`
7. Schaeffer, J., Burch, N., Bjrnsson, Y., Kishimoto, A., Mller, M., Lake, R., Lu, P., Sutphen, S.: Checkers is solved. Science **317** (2007) 1518–1521
8. Bagot, J., Anderson, J., Baltes, J.: Vision-based multi-agent slam for humanoid robots. In: Proceedings of CIRAS-2008). (June 2008) 171–176
9. Baltes, J., Cheng, C.T., Bagot, J., Anderson, J.: Vision-based obstacle run for teams of humanoid robots (demonstrated system). In: Proceedings of AAMAS-2011, Taipei, Taiwan (May 2011) 1319–1320
10. Softbank Robotics: Pepper. `http://www.softbankrobotics.com`
11. Kuroki, Y.: A small biped entertainment robot. In: MHS2001. Proceedings of 2001 Int. Symposium on Micromechatronics and Human Science. (2001) 3–4
12. Koretake, R., Kaneko, M., Higashimori, M.: The robot that can achieve card magic. ROBOMECH Journal **2**(1) (2015)
13. Nuñez, D., Tempest, M., Viola, E., Breazeal, C.: An initial discussion of timing considerations raised during development of a magician-robot interaction. In: Proc. ACM/IEEE Workshop on Timing in Human-Robot Interaction HRI. (2014)
14. Tamura, Y., Yano, S., Osumi, H.: Modeling of human attention based on analysis of magic. In: Proceedings of HRI 2014, New York, NY, USA, ACM (2014) 302–303
15. Morris, K., Anderson, J., Lau, M.C., Baltes, J.: Interaction and learning in a humanoid robot magic performance. In: Proceedings of the AAAI Spring Symposium on Integrated Representation, Reasoning, and Learning in Robotics. (March 2018)
16. Kotsiantis, S.B., Zaharakis, I., Pintelas, P.: Supervised machine learning: A review of classification techniques (2007)
17. Arkin, R.C.: Behavior-based Robotics. MIT Press, Cambridge (1998)
18. Anderson, J., Baltes, J.: An agent-based approach to introductory robotics using robotic soccer. Int. Journal of Robotics and Automation **21**(2) (February 2006)
19. Liu, T., Baltes, J., Anderson, J.: Archangel, a flexible and intuitive architecture for intelligent mobile robots. In: Proc. of CIRAS-2005, Singapore (December 2005)