The Videogame Affordances Corpus

Gerard R. Bentley[1], **Joseph C. Osborn**[1]
[1]Pomona College
gbkh2015@mymail.pomona.edu, joseph.osborn@pomona.edu

Abstract

Videogames are created for human players whose commonsense knowledge of real-world objects and interactions (and their familiarity with other games) primes them for successful play. Action games feature recurring formal elements including a directly controlled avatar, moving enemies, resource pickups, and portals to new map areas; mapping these onto culturally significant symbols helps players learn to play quickly. We present a schema, annotation tool, and dataset for codifying screenshots containing game objects in terms of their affordances, which is suitable for AI agents and machine learning algorithms for a variety of interesting and significant applications.

Introduction

Videogame play from vision has been increasingly successful since the development of Deep Q-Learning (Mnih et al. 2013). While strategy games like *Starcraft* and *Defense of the Ancients 2* can be tackled with substantial computing resources (Vinyals et al. 2019; OpenAI 2018), even relatively simple adventure games like *Montezuma's Revenge* have posed a significant challenge, where superhuman play has been achieved only recently (Ecoffet et al. 2019). This may be in part because adventure games rely on human interpretation of hints and because diverse (but targeted) exploration of the state space is more important than optimizing short sequences.

Even where automated game players have been successful, they are known to be sample-inefficient relative to human players, often requiring lifetimes of practice to achieve their impressive results. While the way in which a machine plays and the way in which a human plays are necessarily different, it seems likely that a major advantage held by humans is their familiarity with cultural signifiers (McCoy et al. 2010) as a powerful prior on the likely outcomes of game object interactions. Humans learn these beliefs through their experiences with real-world objects, socialization, and game literacy—for example, they might suspect a round object can roll, that a skull indicates danger, and that a stationary glowing orb is likely to be beneficial to touch. This suspicion is supported by previous research, which shows that the human advantage collapses when a game's graphics betray player expectations (Dubey et al. 2018). Knowing the interactions that game objects afford, such as "this block will block any agent's movement" or "this key can be picked up by my agent", helps humans guess what they can (and should) do to progress in a game. We define videogame affordances as the actions that agents are capable of performing on objects in the environment. In contrast, an algorithm like Deep Q-Learning or Go-Explore might only implicitly learn to avoid rolling skulls or spiked pits.

Incorporating object affordance data has been helpful for agents that must act in open-ended problem spaces besides games. In the case of improvisational interactions involving a prop, organizing possible actions by the prop's affordances facilitates search and enables stronger performances (Jacob and Magerko 2018). Affordances of real-world objects are determined by their visual and physical properties, and videogame graphics analogously suggest their affordances by genre convention and internal visual consistency. Since we cannot model every game's simulation rules and art direction, we have developed a tool for quickly tagging game screenshots with per-pixel object affordances and built a small corpus of game object affordances. We have exercised this dataset on a prototype model which predicts, from a videogame screenshot, the likely affordances at each pixel location. Applications for this dataset and our model could include:

- Priors for game-playing agents
- Human-legible world representations for Explainable AI
- Transfer learning between games and genres
- Style transfer between games
- Interaction-aware procedural content generation

Schema

Our initial dataset[1] comes from the Nintendo Entertainment System games *The Legend of Zelda* and *Super Mario Bros. 3* (113 and 24 images, respectively), with pixel-wise labels for a reasonably general and complete set of nine affordances (Table 1). We also have broken-out images and affordance data for background graphics and character sprites

[1]https://github.com/gerardrbentley/Videogame-Affordances-Corpus

Solid	Blocks an agent's movement
Movable	An agent can move it
Destroyable	Can be eliminated from the game world
Dangerous	Hurts an agent
Gettable	Can be acquired by an agent
Portal	Reveals more of the game world
Usable	An agent can directly interact with it
Changeable	Can change form
UI	Non-game user interface elements

Table 1: The nine key affordances in our dataset.

appearing in these games, and we are at work coding screenshots from additional games.

We selected our nine affordances based on the authors' expert knowledge of videogame objects and their semantics in NES action-adventure and role-playing games and the first-person shooter *DOOM*. Each affordance is a zero or one label, with more complex objects like breakable doors built out of several individual affordances.

These affordances capture important interactions between player agents and a game environment and seem to generalize to many games involving moving an agent and interacting with environment objects (for example, they seem well-suited to Videogame Description Language (VGDL) games (Thompson et al. 2013)). For example, *solid* objects in a game level usually dictate the path a player agent will follow, while *destroyable* objects may reveal something useful when destroyed. We note that *Changeable* and *UI* are somewhat vague labels compared to the others. The former suggests possibly indirect interactions with the objects, while the latter states that the objects are non-diegetic and exist outside of the world of the agents. A common example of *changeable* objects are closed doors (which are also *solid* and are *portals*) that open under some condition. A door with a lock emblem usually suggests that the player must directly unlock it to change it, which adds the label *usable*. Oppositely, a closed door with no apparent lock often requires the player to perform some action (defeating all enemies, pushing a block in the room somewhere) to change it to an open doorway (non-solid, non-changeable, and a portal).

The primary component of our corpus is the set of annotated screenshots, which are 256×224 images taken from game play alongside binary encodings of each affordance at each pixel. This is the native resolution of NES and SNES games, but the important thing is that the game screenshot and the affordance maps have the same resolution.

Our schema treats tile-based and non-tile-based game screenshots in the same way, but our labeling tool is specialized for tile-based games and it records some extra data (per-tile and per-sprite affordances) for such games. A tool meant for labeling, e.g., 3D game screenshots could work in terms of textures, shaders, or 3D models instead of tiles (Richter et al. 2016).

Annotation Tool

In addition to the annotated screenshots and tiles, we also present the tool we used to process our data. We hope that

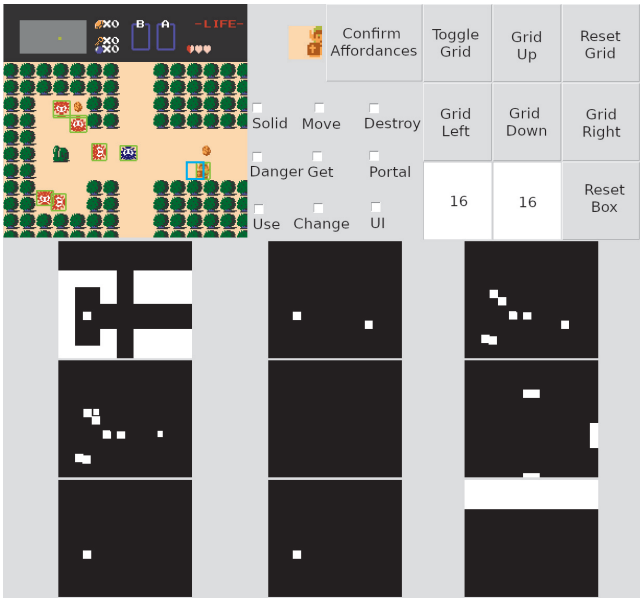

Figure 1: Affordance Annotation Tool. The single blue bounding box notes the current section to be labelled. The nine images correspond to activations of each affordance, where white means the affordance holds

the ready availability of our tooling will support community or crowd-sourced expansion of the dataset.

Tile-based games offer interesting challenges for computer-assisted labeling. Like previous work in annotating game screenshots (Summerville et al. 2016; Guzdial and Riedl 2016), we use OpenCV's template matching (Bradski 2000) to identify sprites and tiles that have already-known affordance labels. Given a new screenshot, our tool (Figure 1) performs an automatic sprite- and tile-wise template matching on a user-adjustable grid. While it is easy (for most NES games) to identify tiles by splitting the image on a uniform 16×16 or 8×8 pixel grid, the underlying grid is not always aligned with the corners of the screen (as in the scrolling *Super Mario Bros. 3*), which requires manual shifting in our process.

Additionally, we need to handle game objects that move off the grid—sprites. Because the number of sprites is usually small compared to the number of tiles, we currently require a spritesheet for each game which an annotator labels in advance. Unfortunately, spritesheets collected by enthusiasts may not exhaust all sprite orientations and may use different color palettes than the emulators used to collect game screenshots (the famous, apopcryphal expansion of *NTSC* is "Never Twice The Same Color"). To make the system more robust to these differences, we use a suite of sprite detectors in a variety of spaces and require that they all match (though at different thresholds): grayscale, RGB, Sobel-derivative, Laplace-derivative, and Canny edges. We believe that a semi-supervised machine learning approach to identify sprites could greatly assist in labelling new games, but we leave it to future work.

After detecting and labelling known tiles and sprites, our

tool asks the user the affordances of unknown tiles in the image, including previously unseen tiles and tiles that are obstructed by sprites (a keyboard-based interface is available for ergonomics). Newly labelled tiles are saved to the collection of known game tiles, but pixels belonging to overlapped tiles are only tagged in the current image. We believe this step could be made more automatic by a system that removes sprites from the background, allowing previously covered tiles to be matched. After this process the image is fully labelled except in locations where sprite detection failed, in which hand annotation is employed.

Applications

Our immediate use-case is to develop a model for predicting game object affordances from vision. This is a multiple-label classification problem—multiple labels may apply to a single pixel and we want the model to predict all of them. The multiple-label setting distinguishes this work from object detection, semantic segmentation, and image classification problems, and it has not been explored in the videogame domain to our knowledge.

Our prototype (a reasonable baseline) adapts the SegNet-Basic architecture (Badrinarayanan, Kendall, and Cipolla 2017) to the multi-label setting: a fully convolutional encoder-decoder network outputs a 9-channel affordance map from a grayscale input image (full RGB data did not significantly improve performance). Figure 2 shows the output of our model for a screenshot from *The Legend of Zelda*. We evaluated our baseline using 3-fold validation, with Hamming loss as the validation metric. The mean Hamming loss in our worst fold was 0.0214 (our best was 0.0189), which is an average of 11,044 mispredicted labels per image (out of a possible 516,096 predictions). On visualizing the worst-labeled images, we believe this is mainly due to the model not seeing relatively rare objects like enemy sprites.

This task is the most direct application of our dataset and supports research in guiding AI game play with human-like priors *without* encoding explicit game- or genre-specific knowledge. Participants in the Generalized Videogame AI competition (Perez-Liebana et al. 2016) have shown that developing domain knowledge of object interactions—similar to that described in our dataset—is a beneficial intermediate goal for agents playing previously unseen games. Some important factors in the decision process of past competition winners include:

- The position of the player agent
- The distance between the player agent and non-player agents
- Identifying non-player agents as friendly or hostile
- Approaching resources if any are present, otherwise a portal
- Exploring unknown areas when other evaluations fail

The VGDL (as used in the competition) represents some of this information explicitly and makes it available to agents. We expect algorithms utilizing this information (even if it is noisy) will reach human performance more easily than algorithms which do not. Our corpus is the first step towards extracting this information from games *in general*.

Feature Representation

The recent Go-Explore algorithm, using only downsampled pixel data, achieved nearly four times the previous best score by current reinforcement learning algorithms in the Atari game *Montezuma's Revenge* (Ecoffet et al. 2019). Supplied with knowledge of the player agent's location and room number, the algorithm outperformed the human record in *Montezuma's Revenge* by an order of magnitude and averages better than human performance in *Pitfall*, a game previously unsolved by strict reinforcement learning. Go-Explore uses a combination of techniques to extract this domain feature information from pixels, including locating the player agent by a pixel color that only appears on that agent's sprite and template matching an image of a key to keep track of where in the game they were found. Our work would help generalize this aspect of extracting domain information from raw pixels to other games. The affordance maps could even be used as a more informative state representation than the coarse visual approximation used by Go-Explore's domain-independent pixel representation. In the same vein, working from affordances rather than from screenshots could promise more transferrable game moment embeddings for search, retrieval, and novelty appraisal (Zhan and Smith 2018; Zhan, Aytemiz, and Smith 2018).

Cutting edge research in image captioning has focused on using neural network architectures to generate accurate natural language descriptions of real-world images (Bai and An 2018). Successful methods include encoder-decoder, attention-guided, and compositional architectures, which operate by deriving context and feature vectors from images via convolutions. Limited research has been done on captioning videogame screenshots and models trained on real-world images fall short (Fulda et al. 2018). We feel that semantic captioning of videogame screenshots is a natural extension of this work, which could also lead into automatic tutorialization (Green et al. 2018).

The predictive model described in the beginning of this section is, in some sense, learning to *see* game screenshots in instrumental terms. The convolutional part of this trained network could be used to bootstrap other models that work from game screenshots, and the final outputs can serve as a transferrable, lowest-common-denominator way for an algorithm to understand a picture from a game.

In addition to generating descriptions of whole images, we see this work fitting with research in more explainable AI systems. Training agents with this high-level game information like affordances (rather than raw pixels) could expose how perceived image features influence the agent's choices. Explaining and observing actions in terms of affordances shows what functional interactions the agent values, even if its internal processes did work from raw pixels.

PCGML

Any game dataset suggests immediate applications in procedural content generation via machine learn-

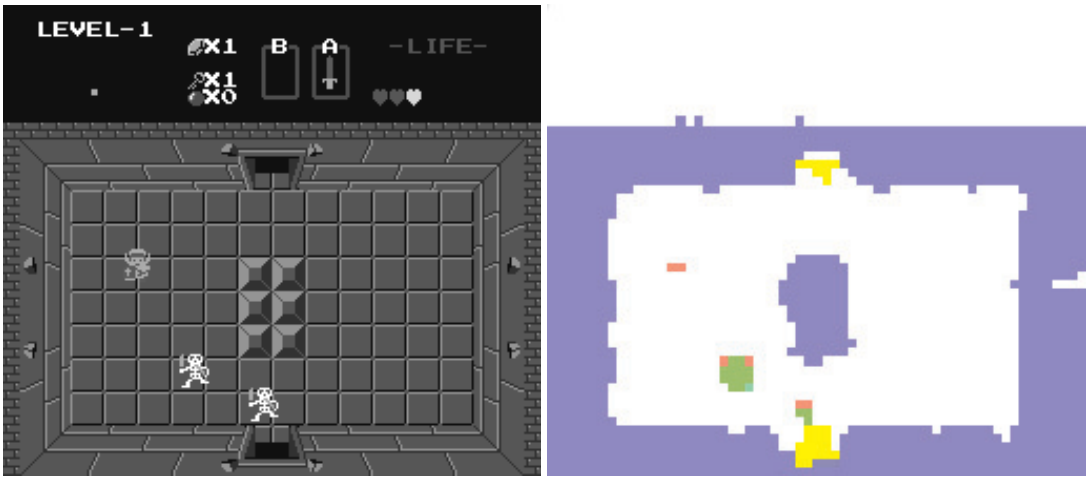

Figure 2: A screenshot from *The Legend of Zelda* (left) and non-zero affordance predictions (right). Predictions for *solid* (Purple), *portal* (Yellow), *dangerous* (Green) and *destroyable* (Red). *UI* area is predicted, but excluded here for clarity

ing (PCGML) (Summerville et al. 2018). Using the model described earlier, it is easy to imagine a level designer sketching out an affordance map and running the model in a DeepDream-like setting to find an image which optimizes the probability of predicting that particular affordance map (Mordvintsev, Olah, and Tyka 2015). Since we have per-sprite and per-tile affordance data, we could come up with an embedding from 16×16-pixel graphics to affordance labels, and attempt to perform vector arithmetic in affordance embedding space to procedurally generate new game graphics.

We also see this work as complementary to existing efforts like the Videogame Level Corpus (VGLC) (Summerville et al. 2016). One interesting interaction is to obtain and tag level data from the VGLC. We could also use the model described above and an automatic mapping system like Mappy (Osborn, Summerville, and Mateas 2017a) to create new maps and interaction-aware legends to expand the VGLC. Combining both corpora could also provide richer features for PCGML algorithms.

Related Work

The most direct related project is the Videogame Level Corpus, which contains structural level layouts and per-game semantic tags for 12 games. The VGLC format required a combination of hand and computer annotations and static file analysis to complete, and its plain-text labels for tile types range from game-specific (particular enemies in *Super Mario Bros.*) to general (e.g., "solid", "breakable"). Our corpus is complementary, focusing on object affordances and interactions in a universal schema, and working at the level of pixels instead of tiles. Importantly, we are concerned with game screenshots and not game levels, so our schema and use cases are unlikely to be fully reconcilable.

A related recent application of the VGLC is *explainable PCGML* (Guzdial et al. 2018). Using level encodings from the VGLC along with expert-provided design pattern labels allows a generator to justify its creations in human-relevant terms, which is vital in a co-creative mixed-initiative setting.

Coming from the opposite direction of the VGLC, automated game design learning (AGDL) is a broad project which has as its goal the automatic extraction of high-level design elements (Osborn, Summerville, and Mateas 2017b). While AGDL focuses on learning game rules from observation and experimentation, our work abstracts specific rules away to focus on broad classes of interaction, and (for now) requires manual tagging.

Conclusion

Our immediate next step is to expand the corpus, both in terms of depth (screenshots and object coverage in each game) and breadth (more games in different genres, with different art styles). Besides manual annotation, we hope to explore the use of instrumented emulators to identify tiles and sprites (Osborn, Summerville, and Mateas 2017a) and to capture object interactions (Summerville et al. 2017; Summerville, Osborn, and Mateas 2017). We also believe that there are natural semi-supervised learning tasks on this corpus: for example, pasting sprites into a sprite-free image at random locations, or reassembling a game screenshot which has been broken up like a jigsaw puzzle (Noroozi and Favaro 2016). Finally, augmenting our dataset with cultural information in free text could help form a fuller understanding of why game objects seem to afford certain uses.

This work is a first step towards helping computers see games as people do, which seems to be a necessary step towards more sample-efficient game playing algorithms. Even though we have focused on instrumental affordances, we have already seen promising initial results in predicting affordances from screenshots. We have also shown that this affordance-oriented view of game images will be useful in game-playing agents and beyond.

References

Badrinarayanan, V.; Kendall, A.; and Cipolla, R. 2017. Segnet: A deep convolutional encoder-decoder architecture for

image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Bai, S., and An, S. 2018. A survey on automatic image caption generation. *Neurocomputing*.

Bradski, G. 2000. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*.

Dubey, R.; Agrawal, P.; Pathak, D.; Griffiths, T. L.; and Efros, A. A. 2018. Investigating human priors for playing video games.

Ecoffet, A.; Huizinga, J.; Lehman, J.; Stanley, K. O.; and Clune, J. 2019. Go-explore: a new approach for hard-exploration problems.

Fulda, N.; Ricks, D.; Murdoch, B.; and Wingate, D. 2018. Threat, explore, barter, puzzle: A semantically-informed algorithm for extracting interaction modes. In *Workshops at the Thirty-Second AAAI Conference on Artificial Intelligence*.

Green, M. C.; Khalifa, A.; Barros, G. A.; Machado, T.; Nealen, A.; and Togelius, J. 2018. Atdelfi: automatically designing legible, full instructions for games. In *Proceedings of the 13th International Conference on the Foundations of Digital Games*, 17. ACM.

Guzdial, M., and Riedl, M. 2016. Toward game level generation from gameplay videos.

Guzdial, M.; Reno, J.; Chen, J.; Smith, G.; and Riedl, M. 2018. Explainable pcgml via game design patterns.

Jacob, M., and Magerko, B. 2018. Creative arcs in improvised human-computer embodied performances. In *Proceedings of the 13th International Conference on the Foundations of Digital Games*. New York, NY, USA: ACM.

McCoy, J.; Treanor, M.; Samuel, B.; Tearse, B.; Mateas, M.; and Wardrip-Fruin, N. 2010. Comme il faut 2: A fully realized model for socially-oriented gameplay. In *Proceedings of the Intelligent Narrative Technologies III Workshop*, INT3 '10, 10:1–10:8. New York, NY, USA: ACM.

Mnih, V.; Kavukcuoglu, K.; Silver, D.; Graves, A.; Antonoglou, I.; Wierstra, D.; and Riedmiller, M. 2013. Playing atari with deep reinforcement learning.

Mordvintsev, A.; Olah, C.; and Tyka, M. 2015. Inceptionism: Going Deeper into Neural Networks . https://ai.googleblog.com/2015/06/inceptionism-going-deeper-into-neural.html.

Noroozi, M., and Favaro, P. 2016. Unsupervised learning of visual representations by solving jigsaw puzzles. In *European Conference on Computer Vision*. Springer.

OpenAI. 2018. Openai five. https://blog.openai.com/openai-five/.

Osborn, J.; Summerville, A.; and Mateas, M. 2017a. Automatic mapping of nes games with mappy. *Proceedings of the International Conference on the Foundations of Digital Games - FDG 17*.

Osborn, J. C.; Summerville, A.; and Mateas, M. 2017b. Automated game design learning. *2017 IEEE Conference on Computational Intelligence and Games (CIG)*.

Perez-Liebana, D.; Samothrakis, S.; Togelius, J.; Schaul, T.; Lucas, S. M.; Coutoux, A.; Lee, J.; Lim, C.; and Thompson, T. 2016. The 2014 general video game playing competition. *IEEE Transactions on Computational Intelligence and AI in Games*.

Richter, S. R.; Vineet, V.; Roth, S.; and Koltun, V. 2016. Playing for data: Ground truth from computer games. In *European conference on computer vision*. Springer.

Summerville, A. J.; Snodgrass, S.; Mateas, M.; and Ontan, S. 2016. The vglc: The video game level corpus.

Summerville, A.; Behrooz, M.; Mateas, M.; and Jhala, A. 2017. What does that?-block do? learning latent causal affordances from mario play traces. In *Workshops at the Thirty-First AAAI Conference on Artificial Intelligence*.

Summerville, A.; Snodgrass, S.; Guzdial, M.; Holmgard, C.; Hoover, A. K.; Isaksen, A.; Nealen, A.; and Togelius, J. 2018. Procedural content generation via machine learning (pcgml). *IEEE Transactions on Games*.

Summerville, A.; Osborn, J.; and Mateas, M. 2017. Charda: Causal hybrid automata recovery via dynamic analysis. *arXiv preprint arXiv:1707.03336*.

Thompson, T.; Ebner, M.; Schaul, T.; Levine, J.; Lucas, S.; and Togelius, J. 2013. Towards a video game description language. *Dagstuhl Follow-ups*.

Vinyals, O.; Babuschkin, I.; Chung, J.; Mathieu, M.; Jaderberg, M.; Czarnecki, W. M.; Dudzik, A.; Huang, A.; Georgiev, P.; Powell, R.; Ewalds, T.; Horgan, D.; Kroiss, M.; Danihelka, I.; Agapiou, J.; Oh, J.; Dalibard, V.; Choi, D.; Sifre, L.; Sulsky, Y.; Vezhnevets, S.; Molloy, J.; Cai, T.; Budden, D.; Paine, T.; Gulcehre, C.; Wang, Z.; Pfaff, T.; Pohlen, T.; Wu, Y.; Yogatama, D.; Cohen, J.; McKinney, K.; Smith, O.; Schaul, T.; Lillicrap, T.; Apps, C.; Kavukcuoglu, K.; Hassabis, D.; and Silver, D. 2019. AlphaStar: Mastering the Real-Time Strategy Game StarCraft II. https://deepmind.com/blog/alphastar-mastering-real-time-strategy-game-starcraft-ii/.

Zhan, Z., and Smith, A. M. 2018. Retrieving game states with moment vectors. In *Workshops at the Thirty-Second AAAI Conference on Artificial Intelligence*.

Zhan, Z.; Aytemiz, B.; and Smith, A. M. 2018. Taking the scenic route: Automatic exploration for videogames. *arXiv preprint arXiv:1812.03125*.