tile2tile: Learning Game Filters for Platformer Style Transfer

Anurag Sarkar and Seth Cooper

Northeastern University
sarkar.an@northeastern.edu, se.cooper@northeastern.edu

Abstract

We present *tile2tile*, an approach for style transfer between levels of tile-based platformer games. Our method involves training models that translate levels from a lower-resolution sketch representation based on tile affordances to the original tile representation for a given game. This enables these models, which we refer to as filters, to translate level sketches into the style of a specific game. Moreover, by converting a level of one game into sketch form and then translating the resulting sketch into the tiles of another game, we obtain a method of style transfer between two games. We use Markov random fields and autoencoders for learning the game filters and apply them to demonstrate style transfer between levels of Super Mario Bros, Kid Icarus, Mega Man and Metroid.

Introduction

Machine learning-based methods have come to be increasingly applied for creative applications, particularly in the domain of visual art. One such popular application has been neural style transfer—the use of convolutional neural nets for rendering the content within an image in various styles. Since being introduced by Gatys, Ecker, and Bethge (2015), a large body of research has centered around improving and experimenting with various methods for style transfer (Jing et al. 2019) as well as developing user-facing style transfer tools (Champandard 2016). Models such as CycleGAN (Zhu et al. 2017) and pix2pix (Isola et al. 2017) have become especially popular for their ability to learn translation functions between sets of images, thereby enabling a wide variety of artistic applications and installations.

Relatedly, a subset of Procedural Content Generation via Machine Learning (PCGML) (Summerville et al. 2018) methods have come to focus on more creative applications of ML beyond just generating levels for a given game. These have centered around reasoning about the design spaces of multiple games taken together and have involved domain transfer (Snodgrass and Ontañón 2016) and combinational creativity-based methods for content generation (Guzdial and Riedl 2018b; Sarkar et al. 2020). Sarkar and Cooper (2020) recently proposed the term Game Design via Creative Machine Learning (GDCML) to classify this creative subset of PCGML, further calling for taking inspiration

from existing creative ML approaches in visual art and music to inform new creative ML approaches for games. One such application is style transfer for games. Despite the recent rise of GDCML works, videogame style transfer has remained underexplored. Traditional style transfer methods in the visual art and image domains are pixel-based and thus not well suited for games. In having to satisfy functional constraints, games in several PCGML works have typically benefited from precise, discretized tile-based representations. Thus, a style transfer method suited for games could also utilize such tiled representations.

In this work, we introduce *tile2tile*, an approach for style transfer between tile-based platformer games. Our method is inspired by pix2pix but instead of learning to translate between sets of images, we learn to translate between sets of tile-based levels, hence the name. More specifically, we train models to convert levels from a game-agnostic form based on tile affordances to game-specific tile representations. Then by converting a level from game A to the sketch form and then using the model to convert the resulting sketch form to a level from game B, we obtain a method for transferring style between games. We demonstrate our approach using Markov random fields and convolutional autoencoders to transfer style between levels of the platformer games *Super Mario Bros.*, *Metroid*, *Mega Man* and *Kid Icarus*. This work thus contributes:

1. An approach for style transfer between tile-based games
2. A demonstration of the approach using Markov random fields and convolutional autoencoders

Background

Neural style transfer was introduced by the seminal work of Gatys, Ecker, and Bethge (2015) and has since been the subject of a wide variety of research, with a survey given by Jing et al. (2019). While traditionally this referred to transferring styles of individual images, the pix2pix model (Isola et al. 2017) enabled the notion of learning translation functions between sets of images by training on pairs of images, in turn allowing for style transfer-adjacent applications between different image domains. We use a similar paired approach for training our models, but operate in the tile domain of game levels rather than the pixel domain of images. To our knowledge, the only prior work attempting style transfer within games is the work of Dadfar, Huang, and
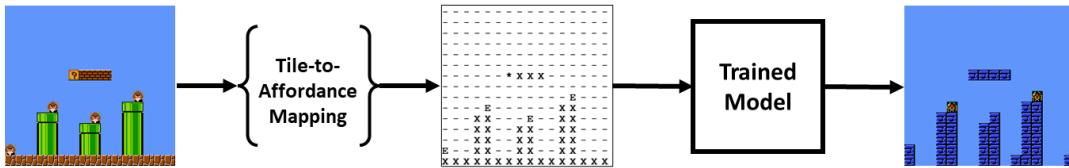
Figure 1: tile2tile style transfer pipeline. In this example, an original Mario level is converted to sketch form using a tile-to-affordance mapping and then a trained model converts the sketch to a Metroid level.

Çelik (2021) and a demo by Google Stadia (Google 2019) but both focus only on transferring visual image style on to the game without considering affordances and are pixel-based rather than tile-based.

Related to style transfer, a key focus of recent creative PCGML research has been to learn, reason about and leverage design spaces spanning a number of games, rather than learn distributions of individual games. This has involved generating new types of content by recombining learned game graphs (Guzdial and Riedl 2018a), blending latent representations (Sarkar, Yang, and Cooper 2019), learning affordance-based tile embeddings across multiple games (Jadhav and Guzdial 2021) and performing domain adaptation (Snodgrass and Ontañón 2016). The last of these is particularly related to our work but differs in learning direct mappings between the levels of two games and focusing on generating new levels for the target domain, whereas we learn models that map a unified affordance-based representation to the tiles specific to a game and focus on transferring the style of a given level. Additionally, we refer to levels in the affordance-based representation as sketches, borrowing vocabulary from (Snodgrass 2019; Snodgrass and Sarkar 2020) who use sketches to refer to lower-resolution level representations indicating only solid and non-solid tiles. The set of affordances we use is a simplified subset of those used by Sarkar et al. (2020) to blend levels of multiple platformers. Their tile-based affordances were in turn inspired by the pixel-based affordance work of Bentley and Osborn (2019). Recent game blending work (Sarkar et al. 2020; Sarkar, Yang, and Cooper 2020) i.e. generating new games by blending levels/mechanics of existing ones, is related to style transfer but differs in that it focuses on blending together different games where as we learn to map from one game to another. Closely related to our work is also the recent work of Chen et al. (2020) who generated Mario levels using an input image, thus performing a version of style transfer. Our work differs in attempting style transfer between game levels. Similar to them, we also use Markov models and autoencoders, both of which have been used in several prior PCGML works. Snodgrass and Ontañón (2017) used both Markov chains and Markov random fields for generating platformer levels while autoencoders have been employed for level repair (Jain et al. 2016) as well as learning level design patterns (Guzdial et al. 2018).

Method

Overview The overall tile2tile approach is depicted in Figure 1. This method of transferring style between levels

of two games A (source) and B (target) consists of:
1. Converting the source level from A into a low-resolution representation *(sketch)* based on tile affordances
2. Translating the level sketch into the original tile representation of game B using a trained model *(filter)*

Thus, central to the approach is the concept of tile affordances. In general, affordances (Bentley and Osborn 2019) for an object refer to the types of in-game actions and interactions that it permits. The key idea that enables our proposed approach is that while games differ in their specific tile-based representations, they can be considered to share a unified affordance-based representation. For example, goombas appear in Mario while metroid creatures appear in Metroid but both of these are enemies that can harm the player; similarly, Mario features coins while Metroid features various powerups but both can be considered as having the affordance of being collectible. Hence by defining a common affordance-based level representation shared by a set of games, and combining it with a model that maps from affordances to the specific tiles of a game, we obtain a method of style transfer by converting a level from a source game to the affordance representation and applying the model for the target game to map this affordance representation to the original tiles of the target game.

We can view this notion of affordances and tiles as analogous to the classic style transfer concepts of *content* and *style* where content refers to the underlying structures and objects within an image while style refers to the high-level visual characteristics such as color, paint style, textures etc. Similarly, we can view affordances as representing the underlying game-agnostic structures and topology such as generic solid tiles and collectibles (i.e. the content) and the tiles as representing the high-level game-specific attributes such as pipes in Mario and powerups in Metroid (i.e. the style). Thus, under this conception of style transfer, when transferring style from one game to another, our goal is to keep the underlying affordances similar while transforming the high-level game-specific tiles.

To perform this, for each game, we define a fixed mapping that translates each of its original tiles to a tile in the common affordance representation, shared across all games. By applying this mapping, we obtain lower-resolution versions of the original levels for each game, which we refer to as *sketches*. Then, for each game, we train a model to convert the game's sketches back to its original tile representation. We refer to these models as *filters* for their respective games, analogous to image filters which similarly transform the visual characteristics of images. Since each game's fil-
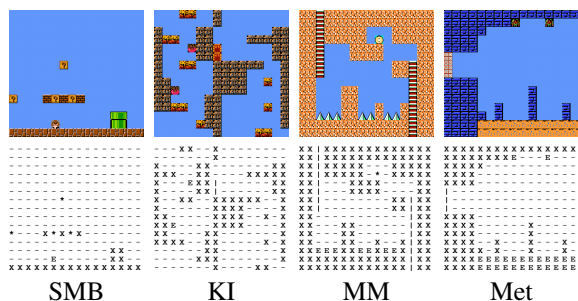
Figure 2: Example segment from each game shown using original tiles and corresponding sketch representation.

Aff	SMB	KI	MM	Met
X				
E				
|	None			
⋆		None		None

Table 1: Tile-to-affordance mapping. Background (−) was the same for each game.

Figure 3: Markov random field training process. Green context sketch tiles were used to predict red game tile.

ter converts a sketch into that game's tile representation, we hence obtain a style transfer process by converting a level from a source game into a sketch and then applying the target game's filter on that sketch to obtain the same source level but in the target game's tile representation. In this work, we implement filters using Markov random fields and convolutional autoencoders.

Tile and Affordance Data We used levels of 4 platformer games—*Super Mario Bros (SMB)*, *Kid Icarus (KI)*, *Mega Man (MM)* and *Metroid (Met)*—all taken from the Video Game Level Corpus (VGLC) (Summerville et al. 2016). VGLC levels are in text format with different characters representing different tiles. Since there is overlap in terms of the same characters being used to represent different tiles across games, we edited the original tilesets to ensure that each game had a completely distinct set of tiles. For the affordance tileset common to all games, we opted to use a much simplified version of the set of affordances used by Sarkar et al. (2020), settling on the following:

- X: *solid* (e.g., ground, platforms, breakable)
- |: *passable, climbable*, (e.g., doors, ladders)
- E: *hazard, enemy*, (e.g., goombas, spikes, lava, metroids)
- ⋆: *collectable*, (e.g., coins, powerups)
- −: *empty*, (e.g. background)

Example levels in their original and affordance representations are shown in Figure 2. For each game, we defined a hand-authored mapping from each original tile in the game to one of the above affordances. The mapping is shown in Table 1. The number of distinct tiles in SMB, KI, MM and Met were 14, 6, 16 and 8 respectively. Since the number of distinct affordance tiles in this work is only 5, the tile-to-affordance mapping is many-to-one and hence this translation can be done manually, as the conversion is deterministic. However, each affordance tile could map to several
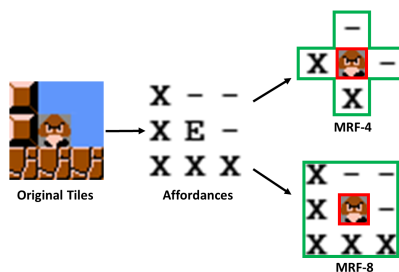
game tiles and thus we need a model that can appropriately translate affordance sketches to levels for a particular game. For this, we test two approaches—Markov random fields (MRFs) and convolutional autoencoders. The MRFs were trained directly using whole levels from the VGLC. Autoencoders however work with fixed-size inputs and outputs. Since VGLC levels vary in size and are limited in number, we trained the autoencoders on segments of levels, as is typical in most neural net-based PCGML work. For this, we extracted 15x16 segments from each of the 4 games. This was determined based on MM and Met having levels whose horizontal and vertical sections are 15 tiles high and 16 tiles wide respectively. KI levels are also 16 tiles wide while SMB levels are originally 14 tiles high so we added a row of background tiles as padding at the top of each level. Each tile in a segment was converted to a one-hot encoding. Thus, the input dimensions for a segment were nx15x16 where n is the number of unique tiles in that game. To extract segments, we slid a 15x16 window across each level, one row and column at a time, giving us 2643, 1171, 3118 and 3762 segments for SMB, KI, MM and Met respectively. For Met, we extracted segments only from level 3 of the VGLC.

Markov Random Field Our first approach to training filters involves using markov random fields (MRFs) (Clifford 1990). While markov chains (MCs) learn conditional probability distributions (CPDs) for the current state given one or more preceding states, MRFs learn CPDs for the current state given its surrounding states. Though both MCs and MRFs have been used previously for level generation (Snodgrass and Ontañón 2017), we opt to use MRFs since for the task of style transfer, we already have a source level to begin with and thus have a set of surrounding tiles for each position, unlike when generating levels from scratch where typically we would only have preceding tiles. Using an MRF allows us to consider a given tile's dependencies in all directions. Note that in the proposed tile2tile framework, the role of the model for a given game is to learn to map affordance tiles to tiles specific to that game i.e. convert a sketch to a level for that game. Using an MRF, we model this as learning the probability of a location in the target level being a certain tile given the affordances of the surrounding neighborhood of that same location in the source level. We refer to this neighborhood as the context for the specific central tile. Thus, the MRF for a game learns the probability
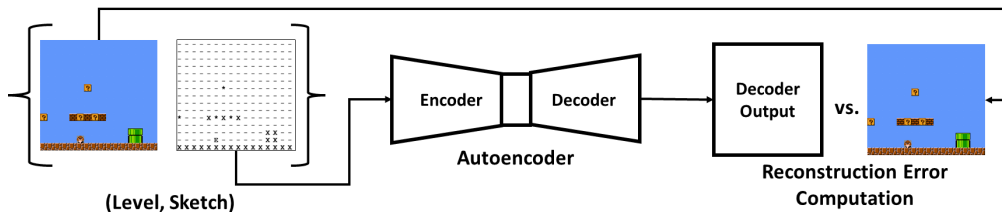
Figure 4: Autoencoder training process.

of tile occurrences given a surrounding affordance context. We test two contexts - one looking at only the 4 affordance tiles north, south, east and west of the central tile (MRF-4) and the other looking at the full 8-tile surrounding context (MRF-8). An example is shown in Figure 3.

Training an MRF model consists of two steps—1) determining absolute counts and 2) estimating probability distributions. We implement MRFs as dictionaries with each unique context serving as a key and each key mapping to the distribution of center tiles for that context. To train MRFs for a given game, we first convert each level to its sketch form using that game's pre-defined tile-to-affordance mapping. Then, we slide a 3x3 window across each level, moving one column horizontally and/or one row vertically at a time, updating the counts of original center tiles (i.e. before converting to sketch form) for each context that we find along the way. Thus, after this step, the dictionary stores the number of times each original game tile appears in the center for each unique sketch context. In the final step, we convert each context's absolute counts into a distribution that we can sample from when performing style transfer. After training, we can use the trained MRF for a game T to change the style of levels from other games to that of T using the pipeline shown in Figure 1. That is, we convert the source level into sketch form using the tile-to-affordance mapping and then similar to training, slide a 3x3 window across the level and replace the center tile by sampling from the trained MRF. More specifically, we look up the MRF's dictionary using the context and then sample a tile according to the learned distribution. If the context is not found (i.e. was not seen during training), we randomly sample a tile with the same affordance as the current center tile in the source level. Example levels are shown in Figures 7 and 8. Note that for MRFs, the style transfer process by design explicitly keeps the affordances fixed from source to target.

Autoencoder Our second approach to learning filters is through the use of convolutional autoencoders. Autoencoders (Hinton and Salakhutdinov 2006) consist of encoder and decoder neural networks. The encoder maps the input data to a compressed, hidden representation which is then used by the decoder to learn to reconstruct the original input in an unsupervised manner. Convolutional autoencoders feature encoders and decoders made up of convolutional rather than fully-connected layers. For training the autoencoders for each game, we collected pairs of level segments from that game and their corresponding sketch representations. We then used the sketches as the encoder inputs but computed

the loss on the decoder outputs using the original segment. Thus, through training, the model learned to convert a sketch of a segment to its original tile representation. This training process is depicted in Figure 4. Our approach is analogous to denoising autoencoders (Vincent et al. 2008) where the encoder input is a noisy form of the original and loss is computed between the decoder output and the original input. We use a translated input where instead of adding noise, we translate to affordances. The underlying structures in inputs/target outputs are identical with only the representation differing. After training, the autoencoders can perform style transfer similarly to MRFs. We take a 15x16 segment of the source game, convert it into sketch form, forward it through the layers of the autoencoder for the target game which finally outputs the segment translated to the tiles of the target game. Example levels are shown in Figure 6. In contrast to the MRF, the process does not keep the affordances fixed explicitly since the autoencoder works at the segment-level rather than at the tile-level. We use a convolutional autoencoder implemented using Pytorch (Paszke et al. 2017). The encoder and decoder consisted of 3 convolutional and 3 transpose convolutional layers respectively, with both using Batch-Normalization and ReLU activation. The encoder inputs were of dimension 5x15x16(=1200) with the 5 channels corresponding to there being 5 unique affordance tiles. The decoder outputs were of dimension (nx15x16) where n varied based on the number of unique tiles in the particular game. Using the format (in_channelsxout_channels, kernel_size, stride_length), the encoder layers were (1200x512, 4, 1), (512x256, 4, 1) and (256xhidden_size, 4, 2) while the decoder layers were (hidden_sizex64, 4, 2), (64x128, 4, 1), (128x(nx15x16), 4, 1). Models were trained for 250 epochs using binary cross entropy loss and the Adam optimizer with an initial learning rate of 0.001, decayed by 0.1 each time the training loss hit a plateau for 50 epochs. We experimented with several hidden sizes finding 128 and 256 dimensions to most consistently capture the affordance structures of the source game. For the rest of the paper, we thus use the 256 (AE-256) and 128-dimensional (AE-128) versions.

Evaluation

Due to the subjective nature of style and aesthetics, evaluating style transfer is a challenging problem (Jing et al. 2019). Ideally we could perform a qualitative user study and elicit opinions from observers regarding the goodness of style transfer but for this initial work, we perform a quantitative evaluation focusing on affordances and tiles.
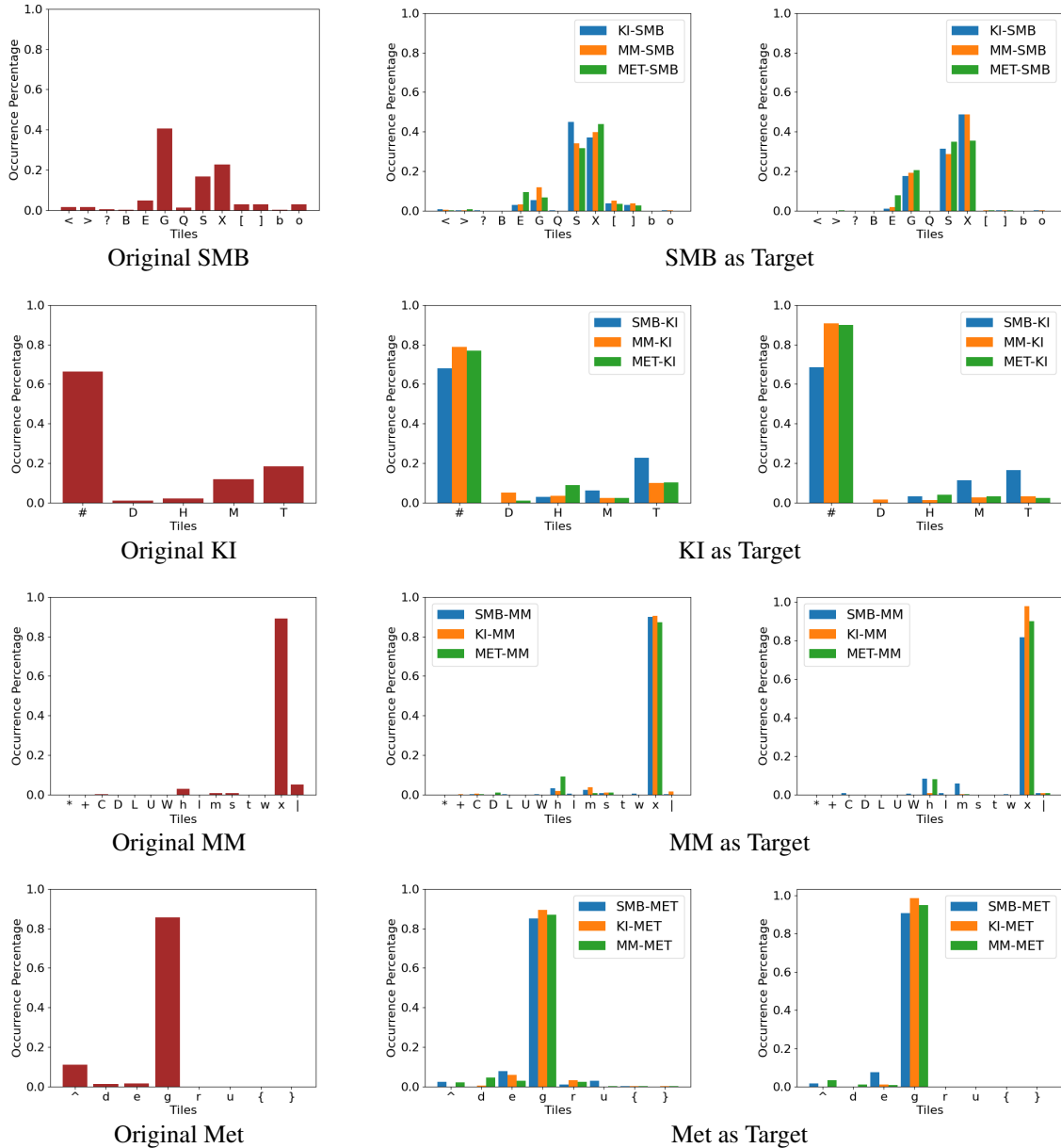
Figure 5: Tile histograms for each original game (left) and that game as the target using MRF-4 (center) and AE-256 (right).

Content As mentioned before, style transfer is often described in terms of *content* and *style*. Typically, the goal is to maintain the content of the source image while applying the desired new style in the target image. This distance between the content of the source and target image is the content loss. Analogously, when performing style transfer on a level from a source to a target game, we want the underlying affordances related to the overall level topology and structure (i.e. content) to be close to the source while changing the higher-level tile representations (i.e. style) to that of the target. Thus, we compare the tile affordance patterns in the input source level and the output target level. For this, we use the Tile Pattern KL-Divergence metric (Lucas and Volz 2019) but use the affordance tiles, not the original game tiles. To avoid confusion, we refer to this as the Affordance Pattern KL-Divergence metric (APKLDiv) in this work. Thus, APKLDiv can be viewed as being analogous to the content loss from visual art style transfer. For MRFs, since the style transfer process by design keeps the affordances fixed from source to target, this evaluation would not be useful. However, it is a useful evaluation for autoencoders since they have to learn to keep the affordances the same.

To perform this evaluation, for every pair of games S (source) and T (target), we apply T's autoencoder on all orig-

Source-Target	TF-Target vs OG-Source	TF-Target vs OG-Target
KI-SMB	**0.71 ± 0.61**	1.52 ± 1.36
MM-SMB	**1.39 ± 1.15**	2.13 ± 1.70
Met-SMB	**1.32 ± 1.19**	2.05 ± 1.63
SMB-KI	**0.27 ± 0.22**	0.87 ± 0.58
MM-KI	**1.34 ± 1.11**	1.99 ± 1.6
Met-KI	**1.17 ± 1.02**	1.72 ± 1.38
SMB-MM	**0.45 ± 0.32**	1.08 ± 0.62
KI-MM	**0.38 ± 0.36**	1.8 ± 0.56
Met-MM	**0.82 ± 0.61**	1.23 ± 0.98
SMB-Met	**0.19 ± 0.16**	1.22 ± 0.74
KI-Met	**0.38 ± 0.38**	2.31 ± 2.14
MM-Met	**0.97 ± 0.74**	1.67 ± 1.45

Table 2: Mean APKLDiv values along with standard deviation using AE-256 for each source-target pair. TF-Target refers to levels obtained by transferring style from original source (OG-Source) levels. Values obtained on comparing with original target (OG-Target) levels are also shown.
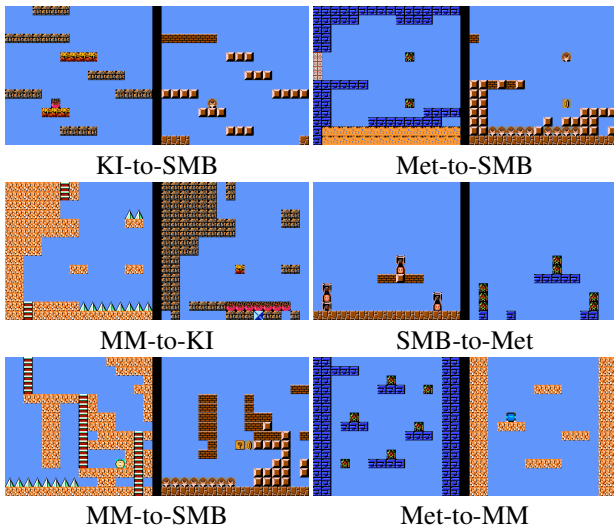

KI-to-SMB   Met-to-SMB

MM-to-KI   SMB-to-Met

MM-to-SMB   Met-to-MM

Figure 6: Example style transfers using autoencoders

inal segments of S. This gives us 2 sets of segments—the original segments of S and the generated segments of the target game T. We then compute the APKLDiv values between these 2 sets using their sketch forms, averaging values obtained for patterns of size 2x2, 3x3 and 4x4. The method for computing these values is given in (Lucas and Volz 2019). An additional, useful set of levels to consider is the original levels for the target game T, also in sketch form. Consider the example of an SMB-to-KI conversion where SMB is the source S and KI is the target T. For effective style transfer, we want the APKLDiv values between the style-transferred KI segments and the original SMB segments to be lower since the goal is to maintain the affordances of the original SMB segments while transferring style to KI. On the other hand, the distance between the style-transferred KI segments and the original KI segments should be higher because the affordances of the transferred levels should be closer to the source SMB segments than the original KI segments. This

shows that the autoencoder is preserving the content/affordances of the source level rather than just generating the levels it was trained on. Note that each autoencoder was only trained on levels for its target game. Results for AE-256 are shown in Table 2. In all cases, the distance between the style-transferred levels and the original source levels is lower than the distance between the style-transferred levels and the original target levels, as we would expect. For space, we omit results for AE-128 but they followed the same patterns except values in both columns were higher (i.e. it did slightly worse than AE-256). Overall, this evaluation shows that the autoencoder for the target game is able to preserve the content of levels from the source game when tasked with performing style transfer. As a concrete example, consider the Met-to-SMB example in Figure 6. The underlying affordances of the SMB level are uncharacteristic of SMB and are more representative of Metroid. Thus, such SMB levels translated from Metroid would be closer to original Metroid than original SMB levels in terms of their affordances.

Style If we consider the underlying affordance-based level patterns and structures to capture the game-agnostic content of a level, then the game-specific style is defined by the game-specific tile representation of those affordance patterns. For example, consider a level from a source game S which is style-transferred to target game T. If we compare this new game T level with an original level from game T, we expect their underlying content (i.e. affordance patterns) to be different since the former was converted from game S. For the style to be consistent, we want the tile distributions to be similar, since they both now represent game T. To evaluate this, we consider the game-specific tile distributions and compare tile histograms between the generated levels obtained via style transfer and the original levels of that game. For each game, we computed the tile histogram (i.e. the discrete distribution of tiles) using all the original levels in the game. Then for each pair of games (S, T) where S is the source and T is the target, we applied the model for T on each level of S and computed the tile histogram using all style-transferred levels. For space we show results for MRF-4 and AE-256 in Figure 5. While not perfect, the tile distributions obtained after style transfer are similar to those of the original levels. We do not expect these to be identical since some amount of difference is a consequence of trying to match the affordances of the source. The biggest difference is observed for the ground G tile in SMB. This may be due to SMB being the only one with separate solid tiles for the ground (G). Additionally, due to having vertical sections, the other 3 games have several segments without a ground row. Hence it is reasonable that when converting from such segments, the resulting SMB segments have lower proportions of ground tiles than the originals.

Playability Finally, we evaluated the playability of style-transferred levels. Though MRFs can work with whole levels, since autoencoders can only work with fixed-size segments, for an even comparison, we evaluated the MRF playability with segments as well. Using whole levels is additionally complicated by the A* agents we use for determining playability requiring specific start and goal tiles to
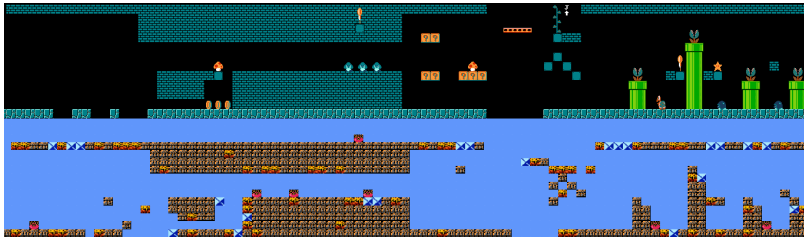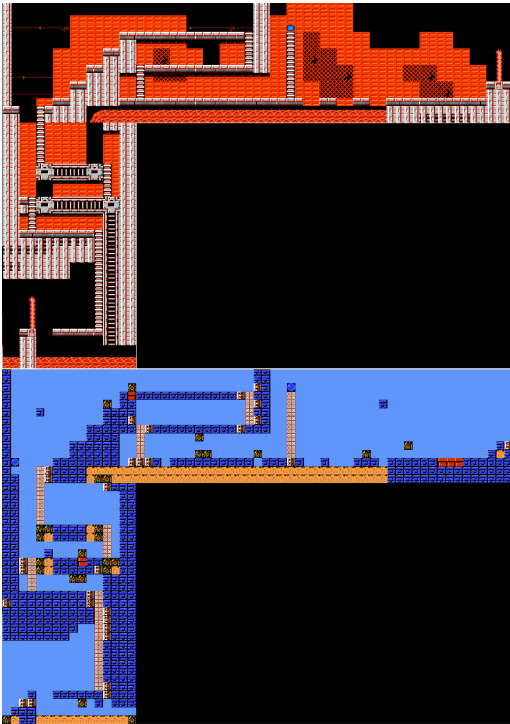
Figure 7: SMB (top) to KI (bottom) using MRF-8

Figure 8: MM (top) to Metroid (below) using MRF-8

	MRF-4	MRF-8	AE-128	AE-256
KI-SMB	75	67.5	**81.25**	75
MM-SMB	39.86	43.36	**80.42**	77.62
Met-SMB	33.07	31.65	72.87	**79.54**
SMB-KI	**75.57**	71.59	62.5	60.8
MM-KI	44.76	46.85	**49.65**	48.25
Met-KI	**48.79**	46.37	32.87	39.31
SMB-MM	**71.34**	68.18	59.09	64.21
KI-MM	**69.23**	63.75	60	67.5
Met-MM	32.63	33.47	38.39	**48.28**
SMB-Met	**85.8**	81.25	60.23	63.07
KI-Met	72.15	**73.75**	61.25	65
MM-Met	46.77	52.45	**61.54**	60.14

Table 3: Percentage of playable segments obtained for all models on each possible pair of games. Highest values per pair are in bold.

be identified, which is not ideal for a Metroid map, due to backtracking. Thus, we evaluated both models using 15x16 segments on which we ran the game-specific tile-based A* agents from (Sarkar et al. 2020) where each game's agent is capable of performing the specific jumps in that game. For each source-target game pair, we applied the model for the target game on every level of the source game and then determined playability by running the target game's agent. A segment is playable if the agent is able to find a path from start to goal. For each segment, we ran the agent in both the horizontal and vertical direction. A segment is playable if either a horizontal or vertical path is found. For each model and source-target game pair, we computed the percentage of source levels that were playable after style transfer. Results are shown in Table 3. We see that performance varies for different game pairs. Autoencoders do best when the target is SMB as well as when the source is MM. MRFs do better when SMB is the source but do poorly when Met is involved in the transfer though it does well for Met-KI. While future

work analyzing playability after style transfer would be useful, it is reasonable that transferring style would lead to a dip in playability and we still obtain reasonably high playability percentages for several pairs. In the future, we could incorporate repair capabilities into the playability agents such as in (Cooper and Sarkar 2020) as well as explore developing generalized agents capable of playing both source and target levels which could in turn lead to generating new types of mechanics.

Conclusion and Future Work

We introduced tile2tile, a style transfer approach for tile-based platformer levels, and demonstrated it using Markov random fields and convolutional autoencoders. In the future, we wish to further validate this approach using user studies and develop an application for users to apply the trained filters to perform style transfer on pre-existing or hand-authored levels. Additionally, we could apply the approach to augment the VGLC by adding style-transferred levels to the corpus and in general use style-transferred levels to supplement levels for games with insufficient training data. A further direction would be to generalize the method to other genres. Our approach should work for tile-based games broadly as long as a common set of affordances can be defined between source and target games e.g. a tile-to-affordance mapping for dungeon-based levels could be used to train a model for translating Zelda levels to maps for some other tile-based dungeon crawler.

References

Bentley, G. R.; and Osborn, J. C. 2019. The videogame affordances corpus. In *Experimental AI in Games Workshop*.

Champandard, A. J. 2016. Semantic style transfer and turning two-bit doodles into fine artworks. *arXiv preprint arXiv:1603.01768*.

Chen, E.; Sydora, C.; Burega, B.; Mahajan, A.; Abdullah, A.; Gallivan, M.; and Guzdial, M. 2020. Image-to-level: Generation and repair. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, 189–195.

Clifford, P. 1990. Markov random fields in statistics. *Disorder in physical systems: A volume in honour of John M. Hammersley*, 19–32.

Cooper, S.; and Sarkar, A. 2020. Pathfinding Agents for Platformer Level Repair. In *Experimental AI in Games Workshop*.

Dadfar, O.; Huang, L.; and Çelik, H. 2021. GAN Theft Auto: Autonomous Texturing of Procedurally Generated Interactive Cities. In *2021 4th International Conference on Computer Science and Software Engineering (CSSE 2021)*.

Gatys, L. A.; Ecker, A. S.; and Bethge, M. 2015. A Neural Algorithm of Style. *arXiv preprint arXiv:1508.06576*.

Google. 2019. Stadia GDC 2019 Gaming Announcement. In *Game Developers Conference*.

Guzdial, M.; Reno, J.; Chen, J.; Smith, G.; and Riedl, M. 2018. Explainable PCGML via game design patterns. *arXiv preprint arXiv:1809.09419*.

Guzdial, M.; and Riedl, M. 2018a. Automated game design via conceptual expansion. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, 31–37.

Guzdial, M. J.; and Riedl, M. O. 2018b. Combinatorial creativity for procedural content generation via machine learning. In *Workshops at the Thirty-Second AAAI Conference on Artificial Intelligence*.

Hinton, G.; and Salakhutdinov, R. 2006. Reducing the dimensionality of data with neural networks. *Science*, 313(5786): 504–507.

Isola, P.; Zhu, J.-Y.; Zhou, T.; and Efros, A. A. 2017. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 1125–1134.

Jadhav, M.; and Guzdial, M. 2021. Tile embedding: a general representation for level generation. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*.

Jain, R.; Isaksen, A.; Holmgård, C.; and Togelius, J. 2016. Autoencoders for Level Generation, Repair and Recognition. In *Proceedings of the ICCC Workshop on Computational Creativity and Games*.

Jing, Y.; Yang, Y.; Feng, Z.; Ye, J.; Yu, Y.; and Song, M. 2019. Neural style transfer: A review. *IEEE transactions on visualization and computer graphics*, 26(11): 3365–3385.

Lucas, S. M.; and Volz, V. 2019. Tile pattern KL-divergence for analysing and evolving game levels. In *Proceedings of the Genetic and Evolutionary Computation Conference*, 170–178.

Paszke, A.; Gross, S.; Chintala, S.; Chanan, G.; Yang, E.; DeVito, Z.; Lin, Z.; Desmaison, A.; Antiga, L.; and Lerer, A. 2017. Automatic Differentiation in PyTorch. In *NIPS Autodiff Workshop*.

Sarkar, A.; and Cooper, S. 2020. Towards game design via creative machine learning (GDCML). In *2020 IEEE Conference on Games (CoG)*, 744–751. IEEE.

Sarkar, A.; Summerville, A.; Snodgrass, S.; Bentley, G.; and Osborn, J. 2020. Exploring level blending across platformers via paths and affordances. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*.

Sarkar, A.; Yang, Z.; and Cooper, S. 2019. Controllable Level Blending between Games using Variational Autoencoders. In *Experimental AI in Games Workshop*.

Sarkar, A.; Yang, Z.; and Cooper, S. 2020. Conditional Level Generation and Game Blending. In *Experimental AI in Games Workshop*.

Snodgrass, S. 2019. Levels from Sketches with Example-Driven Binary Space Partition. In *The 15th AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE)*.

Snodgrass, S.; and Ontañón, S. 2016. An approach to domain transfer in procedural content generation of two-dimensional videogame levels. In *Twelfth Artificial Intelligence and Interactive Digital Entertainment Conference*.

Snodgrass, S.; and Ontañón, S. 2017. Learning to Generate Video Game Maps Using Markov Models. *IEEE Transactions on Computational Intelligence and AI in Games*.

Snodgrass, S.; and Sarkar, A. 2020. Multi-domain level generation and blending with sketches via example-driven bsp and variational autoencoders. In *International Conference on the Foundations of Digital Games*, 1–11.

Summerville, A.; Snodgrass, S.; Guzdial, M.; Holmgård, C.; Hoover, A. K.; Isaksen, A.; Nealen, A.; and Togelius, J. 2018. Procedural content generation via machine learning (PCGML). *IEEE Transactions on Games*, 10(3): 257–270.

Summerville, A.; Snodgrass, S.; Mateas, M.; and Ontañón, S. 2016. The VGLC: The video game level corpus. In *PCG Workshop at 1st Joint International Conference of DiGRA and FDG*.

Vincent, P.; Larochelle, H.; Bengio, Y.; and Manzagol, P.-A. 2008. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*, 1096–1103.

Zhu, J.-Y.; Park, T.; Isola, P.; and Efros, A. A. 2017. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, 2223–2232.