Humanized Computing and Communication
Vol. 2, No. 1 (2021) 1-19
© KS Press, Institute for Semantic Computing Foundation
DOI: 10.35708/HCC1869-148365

Intelligent Generation of Storylines for Immersive Point and Click Games

Federico Maria Cau, Angelo Mereu, and Lucio Davide Spano

University of Cagliari at Via Ospedale 72, 09124, Cagliari, Italy
{federicom.cau,angelo.mereu,davide.spano}@unica.it

Received (12/15/2020)
Revised (01/17/2021)
Accepted (02/16/2021)

Abstract. In this paper, we present an intelligent support End-User Developers (EUDevs) in creating plot lines for Point and Click games on the web. We introduce a story generator and the associated user interface, which help the EUDev in defining the game plot starting from the images providing the game setting. In particular, we detail a pipeline for creating such game plots starting from 360 degrees images. We identify salient objects in equirectangular images, and we compose the output with other two neural networks for the generation: one generating captions for 2D images and one generating the plot text. The provided suggestions can be further developed by the EUDev, modifying the generated text and saving the result. The interface supports the control of different parameters of the story generator using a user-friendly vocabulary. The results of a user study show good effectiveness and usability of the proposed interface.

Keywords: story generation; point and click games; equirectangular images; 360 degrees images; intelligent user interface; .

1 Introduction

The pandemic crisis has changed the way people travel and, as a consequence, how they engage with cultural heritage locations. While in the past there was a high demand for very famous sites, now the presence of crowds is pushing the discovery of minor locations. But traditional cultural institutions are often unable to react to such requests and to provide contents that may attract possible

An earlier version of the paper was presented at the First IEEE International Conference on Humanized Computing and Communication with AI.

visitors, e.g., creating attractive, immersive and engaging stories around the location.

Promoting touristic locations requires creating highly engaging multimedia contents, able to highlight the value of cultural and environmental heritage and, at the same time, engaging and attracting possible visitors. Important locations in the mainstream travelling routes dedicate a part of their budget in creating promotional material to share on the web through websites, social media and other media. Smaller locations have a lower budget and they manage such communication channels in a naïve way: usually the content is produced and edited directly by the cultural operators, exploiting the availability of low-cost hardware and software for authoring them. However, simply creating images, videos and uploading them on the web is not very engaging for potential new visitors. A more interesting approach is using interactive storytelling, connecting the sensors and systems with digital material through a narrative plot and engaging the user with an interactive experience.

Considering the recent pandemic crisis, which has changed the way people travel and how they engage with cultural heritage locations, smaller touristic locations have a new opportunity for getting more visitors: the presence of crowds is pushing the discovery of minor locations. But traditional cultural institutions are often unable to react to such requests and to provide contents that may attract possible visitors, e.g., creating attractive, immersive and engaging stories around the location.

In order to respond to such a need, the PAC-PAC project[1], created a web-based authoring environment for cultural operators without programming experience, whose main idea was to exploit the well-established mechanics of the so-called Point-And-Click (PaC) adventure videogames, simplifying the possible interactions and helping end-user developers (EUDevs) in creating such content. PaC is a particular subtype of adventure videogames, where the player embodies the main character that starts a journey through an interactive world explored in the first-person view. The genre is good for representing virtual immersive experiences simulating remote visits. In their original configuration, the word consisted of a set of predefined frames (both photos or videos), connected with each other. The player navigated through them clicking on the active parts in such frames, which usually maintained consistency in both direction and content with respect to the previous one. Besides the links for the navigation and the exploration of the world, Point-and-Click games contain other interactive elements in the scene: by clicking, dragging and combining specific items, the player can collect or combine objects, activate mechanisms and changing the scene state, solve puzzles or riddles, or unlock further environments to be explored, etc. For instance, the player may collect a key that unlocks a door, which opens an unexplored setting, or he/she may find the proper combination in a keyboard for starting a machine. Considering the simplicity of this game mechanics, the relatively low difficulty in creating images or videos and the advances in the End-User-Development techniques, such a genre is a suitable metaphor for building

[1] https://cg3hci.dmi.unica.it/pacpac-project/

authoring tools unlocking the creation of immersive content for users without development skills [7]. Provided with a proper tool, people working in cultural heritage is able to create immersive experiences and publish them on the web, allowing visitors to take virtual tours and, once arrived in the real place, enjoy a related experience in the real world.

The tool in [7] does not solve a related problem: while it is possible for cultural operators to create the content, they may have problems in identifying how to engage the visitors. In PaC games, this means finding a good plot for the game. In this paper, we explore further tool support for creating the PaC games. We use Artificial Intelligence (AI) techniques (i.e. Neural Networks) for generating small stories that provide ideas for defining the game plot, starting from the images providing the game setting. In particular, in this paper we focus on the generation process starting from 360 degrees images, which are difficult to combine in a neural network pipeline with components working on 2D images. We propose a solution based on identifying objects in the 360 degrees image and automatically extract a 2D snapshot of the object by centring the view on the object bounding box.

The tool suggests different ideas according to different literary genres (adventure, novel, science-fiction etc.). It is possible to control some generation parameters through the tool interface, which affects the generated story, such as the number of the detected object considered and the randomness on their picking criteria. The cultural operator can also further develop the plot, modifying the generated text and saving the result.

The paper is organised as follows[2]: in Section 2 we provide some background and we discuss the related word, in Section 3 we discuss the story generation technique and the management of 360 degrees images, in Section 4 we describe the story suggestion interface, in Section 5 we discuss a user evaluation of the interface and Section 6 concludes the paper and discusses possible extensions of the work.

2 Background and Related Work

2.1 Background

The PAC-PAC project (an acronym for *"Point-And-Click - Patrimonio Ambientale e Culturale"*, Point-And-Click - Environmental and Cultural Heritage) aims fostering the promotion of tourist location through videogames. It provides an authoring environment for people working in tourism promotion for creating Point-and-Click (PaC) games set in real-world locations, for distributing them on the web as entertaining promotion material. The tool has a low threshold, requiring the same effort and knowledge needed for creating web or social network contents. The developed games exploit web technologies that work on desktop, mobile and Virtual Reality (VR) devices. We use 360^o videos for supporting immersive virtual visits, which are relatively simple to create with consumer hardware. Figure 1 shows an example of a point and click game and the overall tool interface.

[2] This paper is an extension of a conference paper by the same authors [2]

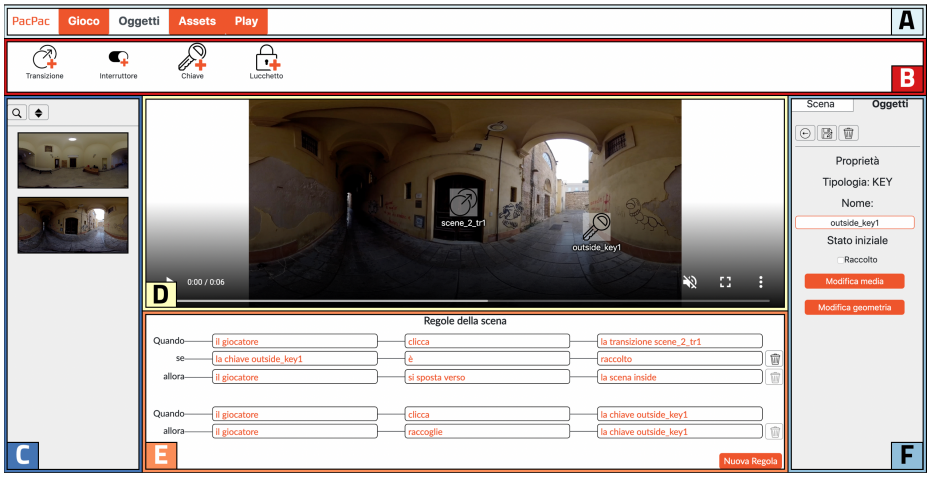

Fig. 1. The PAC-PAC game authoring interface

Basically, the authoring tool supports three main features: i) creating a scene, ii) inserting interactive objects into a scene and iii) defining the behaviour of a scene The authoring environment represents a PaC game as a graph of scenes (videos or images), connected by arcs implementing the transition from a location to another. The interface for editing the game recalls familiar multimedia content editing applications, such as MS PowerPoint. All the scenes are available in the left bar (C), which supports ordering and searching in the list. The main part (D) contains the current scene. It displays a scene and the position of the inner interactive objects. The top-level menu (A in the figure) has four elements: i) Game for adding new scenes and/or modifying experience-level properties (e.g., background music, scene groups), ii) Objects for adding new interactive objects in the current scene, iii) Assets for managing the game assets (videos, overlay images, music, sound files etc.) and iv) Play for testing the game by playing it. When creating a new scene, the user selects a regular or 360^o photo/video from the asset collection, or uploads it from her own files. It is possible to group related scenes using semantic tags, which define macro-locations (e.g., all the scenes related to the same dungeon). Each tag is associated with a specific colour.

The authoring environment provides a set of objects helping the user in defining the game behaviour, displayed in the top bar (B). Each object represents a tool that, added to the scene, introduces an interactive element. For inserting an object into the scene, the user presses the corresponding button in the upper bar. Once inserted, the user modifies its configuration filling the properties panel in the right bar (F). Each object has an interaction area, which receives the player's clicks for activating the object. For instance, the interactive area of a transition is the surface the user clicks for changing the scene. The tool supports creating it by clicking the contour points in the 360o video. In addition, each object has an optional property for animating the interactive area at the object activation through a video (e.g., opening a door when triggering a transition). There are different types of interactive objects besides the transition. For instance, the key

allows unlocking another object when collected, the switch allows changing the current scene configuration according to its status (e.g., a light turns on/off), the counter counts the clicks on the specified interactive areas, the timer triggers an action after the specified time, the locker requires clicking a set of interactive areas in a specified order for triggering an action etc.

The authoring environment uses rules for defining the behaviour of each object in isolation and their interactions for creating the game puzzles. Rules define the dynamic behaviour of a scene, supporting the user in inspecting and controlling the game logic. The rule editor is located in the bottom part of the interface (E). It contains a list of Event-Condition-Action (ECA) rules expressed in natural language according to the following pattern (elements in angular brackets are required, the ones in square brackets are optional):

```
when <subject> <action> <object|value>
[if <condition>* ]
then (<subject> <action> <object|value>)*
```

Once the user completes the game definition, he/she can publish it on the web. Both the authoring environment and the game engine exploit web-based technologies, so players can access games using mobile phones, tablets and personal computers. In addition, the engine supports immersive experiences through Virtual Reality Head Mounted Displays, adapting the input modality to the current device.

2.2 Related Work

In this section, we will analyse the currently available artificial intelligence techniques for generating stories, using perspective and/or equirectangular images as the starting input. In the literature, the research effort in generating text from images focused mainly on an automatic description of the image content. We can find different approaches for labelling the objects inside an image [17] or for providing a good descriptive caption [11].

More recently, the research community put the effort in creating neural networks for generating narrative texts, describing experiences and actions from an image instead of simply listing the identified objects and their characteristics. For instance, Huang et al. [13] introduced a dataset containing 80000 images grouped into 20000 sequences. Each image has a description that considers both the position into the sequence and a story covering all the images in the sequence. Further research exploited the dataset for training AI models, but the main problem in the generation is keeping a coherent plot while generating text that is relevant for all the considered images. All the proposed solutions contain two components: an *encoder*, which extracts the high-level characteristics of the images and a *decoder*, which generates the actual story starting from the encoder input.

The work in [19] investigates such a limitation, proposing a model able to find relationships among non-consecutive images in the sequence. If two consecutive images are very different, the generation process considers the previous ones

that have a higher similarity degree, in order to keep the whole story coherent. In [8] the story generation follows two steps: in the first one, a neural network encodes the content of all images inside a context vector; the second step consists of providing the context and each image in the sequence to five different decoders. Symmetrically, the work in [21] experimented using two encoders and one decoder. The first encoder extracts the features according to the selected sequence, while the second encodes a representation of the sequence itself.

Another important problem to consider while generating the story is the variety of the produced text: two similar images may produce similar text. Hsu et al. [12] apply a penalty to terms already generated using other images, forcing the variation of the story text. Wang et al [22] inserted a further neural network for evaluating the story generation quality. Starting from the representation of the generated text in n-grams (e.g., $n \in [2; 4]$) the network is trained to assign a score to text sub-parts and then to combine them into a global evaluation.

In our work, we started from the approach in [14], which works on a single image producing a longer text if compared with the approached we discussed so far. Regarding the usage of equirectangular images for the story generation, we will train a separate model on an existing dataset made for object detection in equirectangular images by Chou et al. [5], that provides 37 object categories annotated with Bounding Field-of-Views (BFoVs) [23] on real indoor images. We will use the same settings of [5] to train a model on this dataset, using the resulting object detection to create perspective images and use them with the Neural Storyteller model. We will discuss these models and the extension we implemented in Section 3.

3 Plot Generation Engine

The plot generation engine we use in this work is an extension of the Neural Storyteller discussed in [14], which can generate a small romance plotline from a single image. It consists of two components. The first one, described in [24], associates the input image to a set of captions taken from the Microsoft COCO [3] dataset i.e., small sentences describing the image content. It contains two neural networks. The first one is a CNN (convolutional neural network) that detects about 1000 object types. The second is an LSTM (long short-term memory) that maps into a common space both the objects and the sentences.

The second component is an RNN (recurrent neural network) trained on a romance book corpus [25]. It exploits the skip-thoughts [15] model, whose goal is identifying, starting from an encoded sentence, the previous and the following one, according to the text flow in the corpus used in the training phase. The model-induced vectors are called skip-thought vectors. The model consists of three parts:

1. an encoding RNN taking a sentence x_i and returning a fixed-length representation z_i
2. a decoding RNN that, starting from z_i, tries to reconstruct the previous sentences x_{i-1}

3. a decoding RNN that, starting from z_i, tries to reconstruct the following sentences x_{i+1}

In summary, the story generation works as follows. Given the starting image, the CNN (VGG-19 [20]) extracts the features and identifies the objects. Such information goes to the LSTM that puts such information in a space shared with the COCO captions. In such a space, a configurable number of captions are selected (e.g., 15). For passing from the captions to the book-like sentences, the caption is transformed into a fixed-length vector and then the model subtracts a bias vector representing the caption writing style and adds a vector representing the book genre writing style. Such vectors are obtained computing the mean on the skip-though representation respectively on the Microsoft COCO dataset and the romance book corpus. Finally, the biased vector is the input of the decoder that generates the actual story. Internally the model performs a search for finding the most appropriate sentences. It is possible to configure the depth of this search through the parameter beamwidth. This adds more variability to the results starting from similar images.

In our project, we extended the original network for creating the story from more input images and to add more genres (e.g. adventure, sci-fiction etc.). The first extension required to extract the features from a set of images and to identify the most appropriate caption for each of them. After that, we collapse all the captions in a single vector representation, for passing all the information at once to the decoder and obtaining the single story.

For increasing the available literary genres we needed to collect a corpus of books for each genre. We relied on websites offering free books to download (e.g., Project Gutenberg and Smashwords) and we trained the network for generating the following genres:

- Adventure
- Science Fiction
- Fantasy
- Thriller
- Horror

After a preprocessing phase for converting the books into a plain text format and removing the parts that were not interesting for the training (e.g., the table of contents, notes or other additional text) we proceeded to train a decoder for each genre. We kept the same parameters for all of them (e.g., the dimension of the input encoding vector) for having a compositional structure in the underlying AI engine. After that, we proceeded to produce the bias vectors representing the text style for each genre, calculating the mean as described in the generation process.

As already mentioned in section 2.2, we use the 360-Indoor dataset to start dealing with equirectangular images for the story generation. Since this dataset only provides 37 annotated objects commonly found in indoor places, the selection of images is, for the most part, limited to such settings. Nonetheless, we can also use images of outdoor places in which we can find these objects. Despite

having a limited number of objects that may lead to similar sentences in the CNN model, the Neural Storyteller module enriches the context of objects by leveraging the book genres, allowing more variety for the stories. The process we use to generate stories from equirectangular images can be briefly described with the following steps. First, we train a model using the 360-Indoor dataset, obtaining the bounding FoVs for each object detected. After that, we map the equirectangular image into a 3D sphere to get the coordinates of the bounding FoVs and their centroids, projecting the sphere onto the tangent plane. Finally, we crop the planar bounding box of each object, enlarging the field of view to provide some context in which the object is located. We will now analyze each of the previously described steps in depth.

The training configuration is similar to the work in [5]. We first transform the annotated bounding field of views (BFoVs) in conventional bounding boxes with the mapping function used in [4], by projecting the BFoV annotations from spatial coordinates into tangent coordinates. The reason for this transformation is to match the input format of the object detection network employed. We trained a two-stage object detection model on the 360-Indoor dataset, precisely a Feature Pyramid Network (FPN) [18], following the official implementation. We decide to use this model to reproduce the good results obtained in article [5]. Consequently, we used Region of Interest (RoI) align [9] to extract the proposed candidates for second stage, ResNet101 [10] as backbone network pretrained on ImageNet dataset [6] to extract the image features, and the same FPN training parameters.

Once we have the resulting object detection on the target equirectangular image, we transform the resulting bounding FoVs in conventional bounding boxes applying once more the mapping function proposed in [4]. We do this only for presenting conventional bounding boxes to the user, giving them the possibility to consider or not some objects in the story generation by deleting the boxes of non-interesting objects inside the equirectangular images via an interface that we will discuss in the next section. After that, we take as a reference point the centroid of each bounding FoV to generate a perspective projection in the tangent plane, centring the field of view on each detected object.

To warp from the equirectangular panorama to a perspective image we first calculate the 3D coordinates from the target equirectangular image, using the UV mapping process. This methods allow us to map the bounding FoVs coordinates and centroids from 2D (u, v) into 3D sphere map coordinates (x, y, z) according their latitude θ and longitude ϕ (see Figure 2). After that, we project the 360° image to a tangent plane around the sphere according to the bounding FoV centroid recalculated based on the UV mapping, generating our perspective image. Finally, we crop the image containing the object using the bounding FoV coordinates projected to the tangent plane, expanding the coordinates equally by a parameter γ manually set to 20% and tuned on our experiments. We do this in favour of catching some nearby context around the object so that we could have more of them in each cropped image, benefiting the Neural Storyteller module. Since the enlarging procedure takes into account the height and width of the

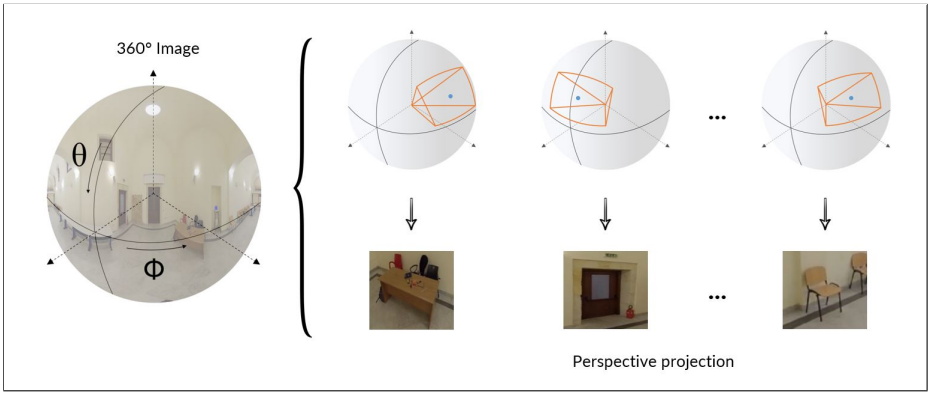

Fig. 2. Converting from equirectangular image to perspective images

equirectangular image projected to the tangent plane, the cropped objects will not exceed the original image boundaries. So, the value of parameter γ might be lower than 20%. Once we have all the object detected represented as perspective images, we can feed them to the Neural Storyteller module.

Once we created both models, we implemented the software services for supporting the story suggestion. We published a web service receiving the URL of the images where the game story is set and it returns the text corresponding to the generated stories. In the service call, it is possible to specify a configuration parameter for modifying the beam-width value and specifying the genres expected in the response.

4 Plot Generation Interface

We integrated the generation interface into a dedicated tab in the main menu bar (see Figure 1-A). Selecting the "Story" tab, the authoring environment shows the interface for managing the storylines depicted in Figure 3. It contains both the stories generated in previous interactions and the button for starting a new generation (the green one).

The story generation sequence is the following. First of all, the user selects a subset of the game-setting background images from the asset list. Once finished, the authoring environment generates a storyline for each genre chosen by the user.

For each selected image, the user can also specify the "object relevance" parameter, which allows controlling the weight of the object detected in that image on the story generation process. Considering the AI model described in Section 3, we map the value specified by the user into the number of captions to consider the specific image. Therefore, if two images have a different value for the object relevance, the one having a higher number will influence more the generation of the entire story plot.

The second parameter the user can control is the story "randomness". In the model in Section 3, it corresponds to the beam width parameter and it indicates how large is the search space for the story text. The wider the space, the

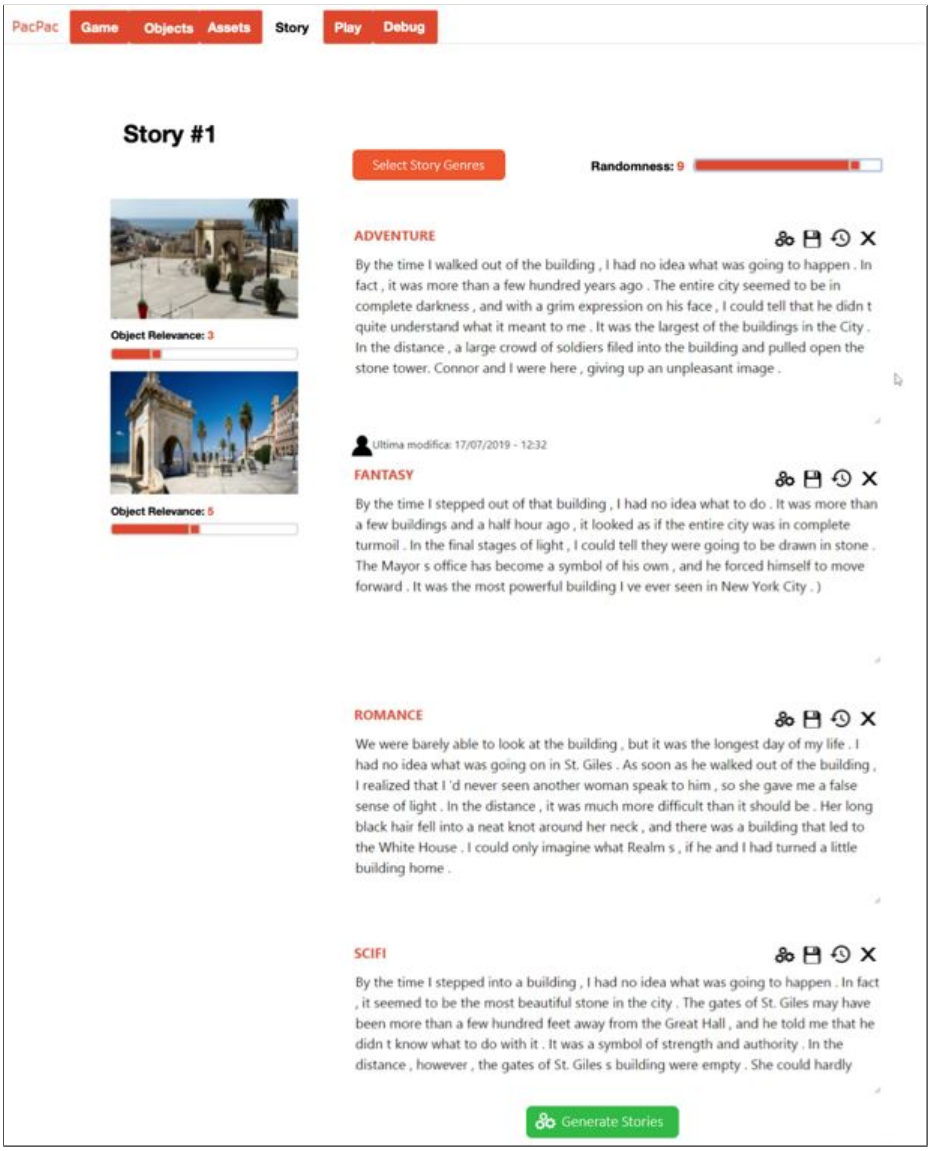

Fig. 3. The story editing interface

more variable the final text will be, but the generation will also take more time. Restricting the search space results in more similar stories for similar images, so the model parameter can be perceived as a randomness feature by the end-user, even if it increases the generation time.

Finally, the last parameter the user can control is the story genre. To save time in stories generation, the environment enables the possibility to select which genres to generate, also allowing to select all of them (see Figure 4). After the stories generation, the user can briefly skim the result and decide to keep all of them or to discard those she is not interested in, by simply clicking the "X" button in the upper-right corner of each genre section (see Figure 3).

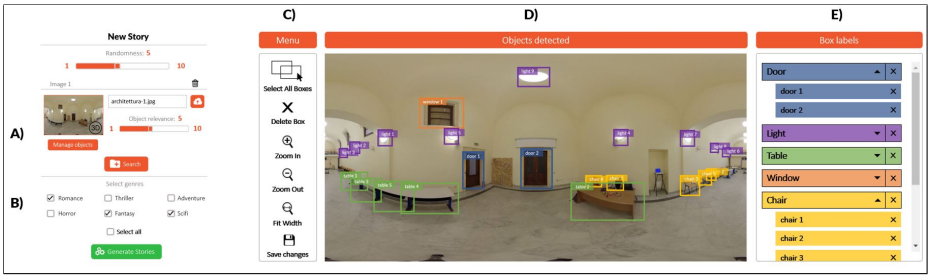

Fig. 4. Managing parameters on equirectangular images the user can control. In part A the user can change the objects of interest by pressing the button "Manage objects" and opening the interface shown on the right side of the figure. In part B, the user can decide which genres she prefers to generate. In part C there is a menu with which the user selects the objects of interest. In part D we have the equirectangular image with the detected objects and in part E a color representation of the classes detected by the model

Regarding the managing of equirectangular images, when the user selects one of these types the environment shows a label indicating the 3D image type and allowing the user to manage the objects she wants to consider during the story generation, by pressing the button "Manage objects". Once the user clicks this button, a pop-up window appears with all the objects detected by the FPN model, and the user can decide those to keep and those to eliminate using both the toolbar menu on the left side and the toolbar label on the right (see Figure 4). The toolbar on the left allows the user to select all the bounding boxes at once, delete the one or more selected boxes, zoom in or zoom out the image, restore the original image size and save the changes made. The toolbar on the right shows a color representation of the classes detected by the model, allowing the user to delete single objects or all the objects inside a class by clicking the "X" symbol beside the class or object name. Once the user has finished, she can decide to save the configuration of the new objects or to revert the changes. The interface also allows managing the objects in equirectangular images after the stories generation.

After the first generation step, we can positively assume that the user might be unsatisfied with the result. The interface allows the user to regenerate a single-story genre by clicking on the gear icon above each story text, or regenerate all the stories genres available by pressing the green button (see Figure 3). If the user wants to modify the available stories genre, she can press the button "Select Story Genres" to the left of the "randomness" parameter (see Figure 3).

Furthermore, the interface supports the update of all the parameters that will trigger again the story generation. For instance, the user can drag the randomness slider and, once released, the interface will update the story text. It is also possible to change the object relevance parameter for each image included in the generation set and, again, the new value will trigger the update of the generated text. Figure 5 shows the effects of the user control on a sample storyline. The generation process considered two images and the adventure genre. In part A, the object relevance for the first image was set to 5, while the second was set to 7. The randomness level is 5. The resulting story is shown in the upper-left part of Figure 5. Then, the user changes the relevance for the first image, setting it to 3. The authoring environment updates the story text, which results in the text in the middle-left part of Figure 5. Finally, the user changes the randomness parameter, setting it to 9. Again, the authoring environment updates the story, which is shown in the bottom-left part of Figure 5.

The authoring environment provides also means for supporting the user in modifying the generated story and adapt it to her purposes, by adding, changing or removing parts of the text. Each storyline is inside a text area. A simple click allows the user to start editing its contents. Once modified, the user can press the save button and the authoring environment will keep the edited version. In the bottom-left part of the box, it shows the information on the last change. However, the user can revert the text to the initial generated version using the reload button next to the "X" one (see Figure 6).

During the game editing, the user can keep a small panel showing the story text in the bottom-left part of the interface in Figure 1.

5 Evaluation

In order to assess the usability and the effectivenesses of the proposed interface, we carried out an evaluation with users.

Each participant read a document containing a brief description on the PAC-PAC project and the story generation feature that was under evaluation. In addition, the document contained the link to the questionnaire after each task and once they completed the test.

The test consisted of the following tasks:

T1 Use the provided credentials for logging in and upload two images for the game setting;
T2 Generate the storylines through the dedicated interface and keep the two genres the participant liked the most. The task description suggested trying using the sliders for controlling the generation process.
T3 Modify the remaining storylines inserting at least a sentence in each one.

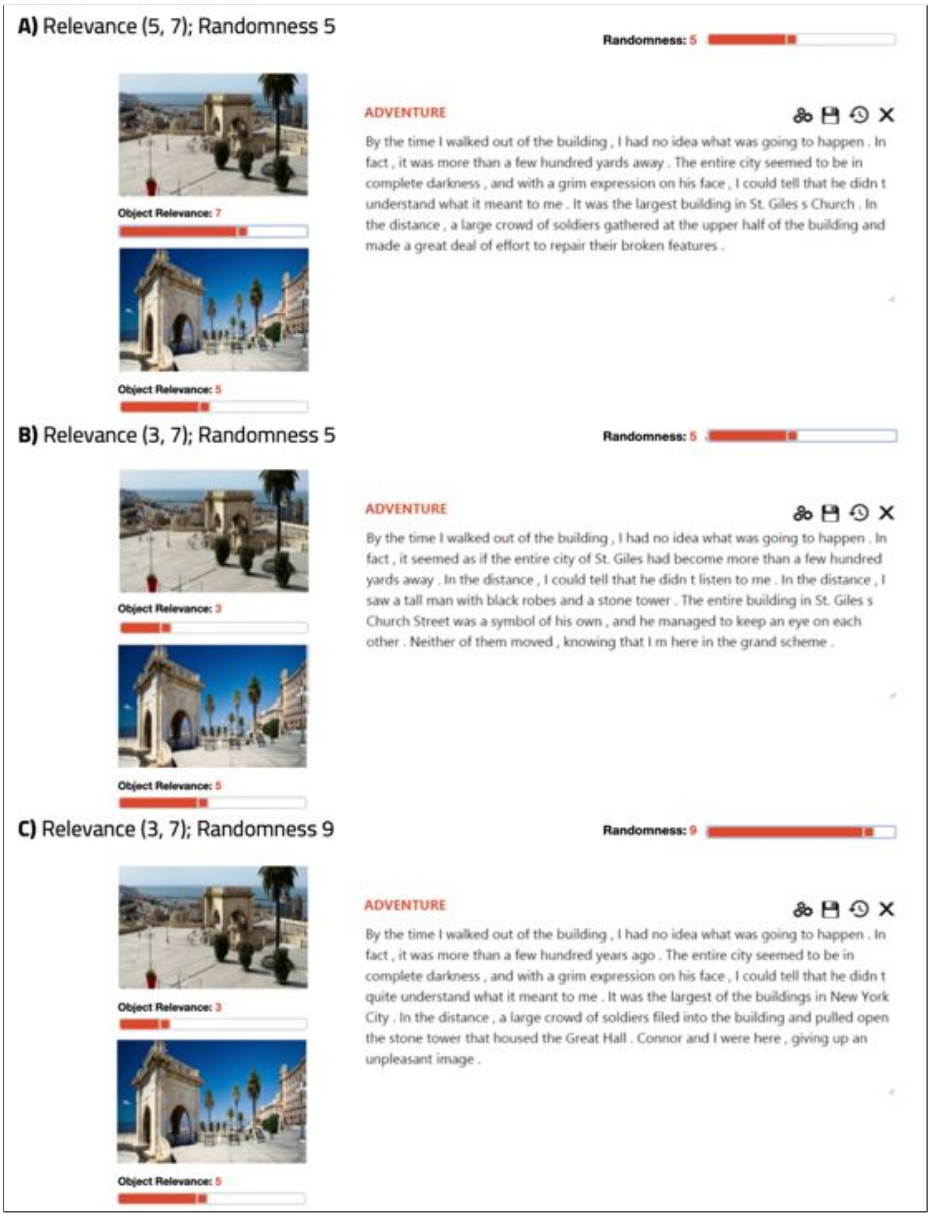

Fig. 5. The effects of the user control on the story generation parameters on a sample adventure plot. The generator considers two images. The starting object relevance are 3 for the first and 5 for the second image. The randomness is 5. From step A to B, the user sets the object relevance for the first image to 3. From step B to C, the user sets the randomness to 9.

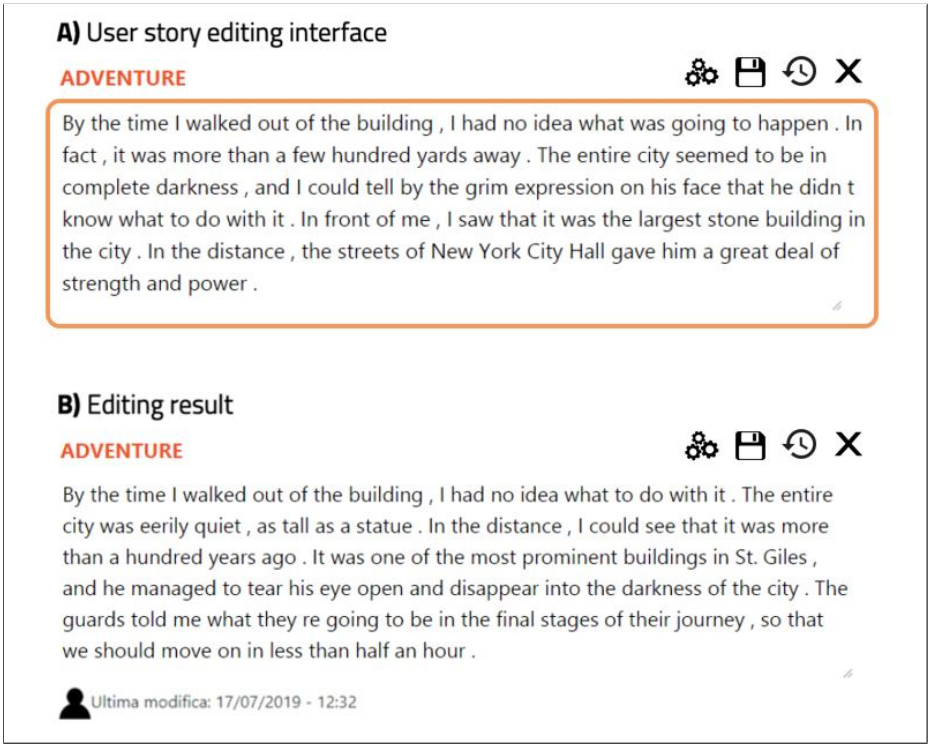

Fig. 6. User interaction for the story editing. In part A, the user modifies the story text by typing the changes in the text area. After pressing the save button, the interface shows the last edit information and the user can eventually use the button to restore the AI-generated story

At the end of each task, we requested the participants to complete the Subjective Mental Effort Questionnaire (SMEQ) [26] measuring the difficulty in completing the task and the After Scenario Questionnaire (ASQ) [16] which measures the ease in completing the task, the adequacy of the time spent and the quality of the provided information.

After completing the test, the participants filled the Software Usability Scale (SUS) [1] measuring the overall usability of the system. In addition, we requested a qualitative rating on the overall usefulness and quality of the generated stories, the correspondence between the images and the generated text and the support for editing the stories. Finally, the participants were invited to express open-ended comments on the application.

Ten people participated in the evaluation, 6 males and 4 females. Four of them had a High school degree, one a Bachelor and 5 a Master degree. They had a good experience with office application ($\bar{x} = 5.2$, $s = 0.63$ on a 1-7 Likert scale) and spend on average 5.3 hours per day on the internet ($s = 1.64$).

All the participants completed all the tasks. The analysis of the SMEQ [26] questionnaire, shows that all task got a rating between 0 and 10 in a 1 to 150

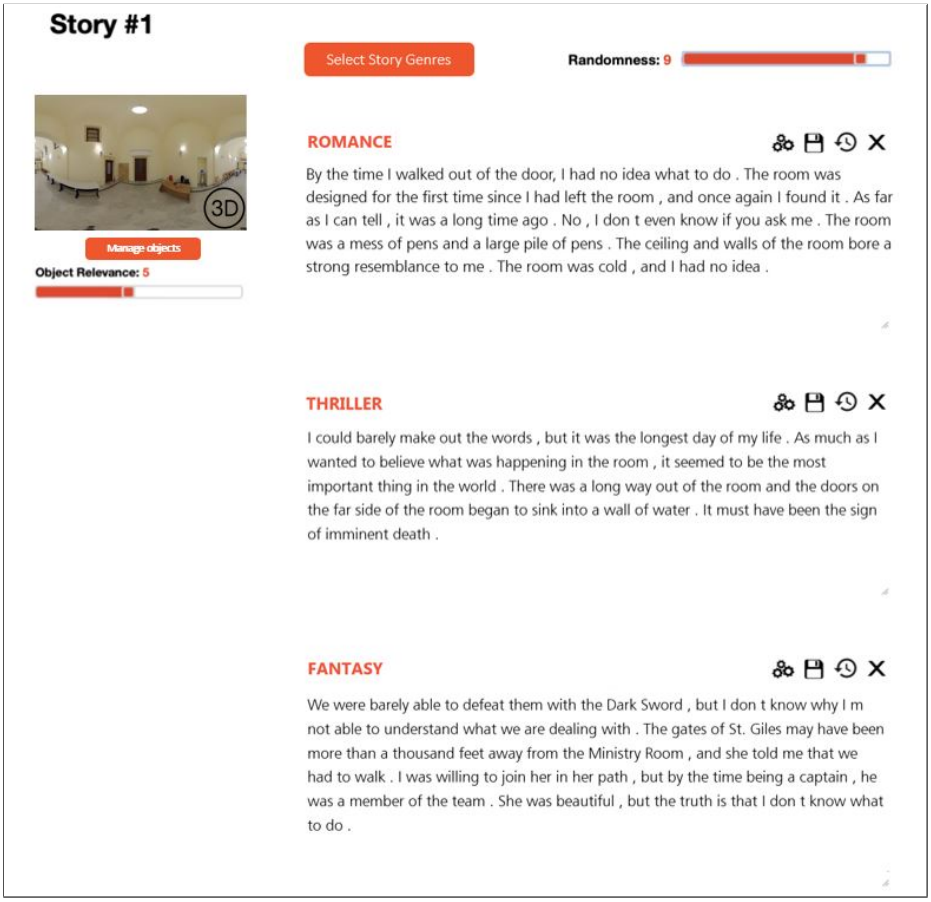

Fig. 7. Stories generated by using an equirectangular image as input

scale. According to the labelling in [26], the tasks are between *Not at all hard to do* and *Not very hard to do*. Figure 8 summarises the test results. We found no significant differences among the tasks. We registered the smaller difficulty value for T1 ($\bar{x} = 1.5$, $s = 3.17$), followed by T3 ($\bar{x} = 3.2$, $s = 4.05$) and T2 ($\bar{x} = 3.3$, $s = 3.3$). From these results, we can conclude that the effort the participant put in interacting with the proposed interface is low.

The ASQ [16] questionnaire results show that the interface was also effective according to the adequacy of the time spent in completing the tasks T1 ($\bar{x} = 0.1$, $s = 0.32$ in a 1-7 Likert scale, the lower the better) and T3 ($\bar{x} = 0.1$, $s = 0.32$). We registered a slightly lower satisfaction for task T2 ($\bar{x} = 0.7$, $s = 0.48$), but that was expected since the generation of the stories for all genres may take a perceivable time if the randomness value is high (maximum one minute) and the users had to wait for the output. Instead, T2 had the best rating on the ease of use ($\bar{x} = 0.3$, $s = 0.48$, in a 1-7 Likert scale, the lower the better). Figure 9 shows a graphical summary of the results for the ASQ [16] questionnaire.

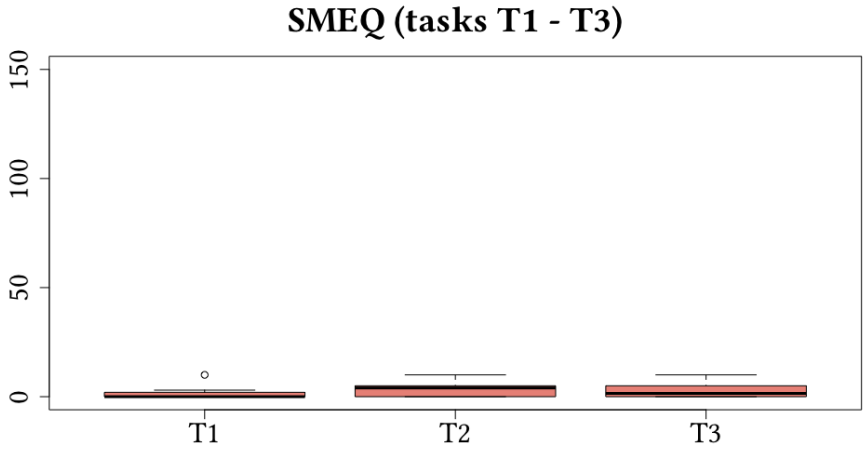

Fig. 8. Subjective Mental Effort Questionnaire (SMEQ) [26] results

The post-test questionnaire confirms that the overall usability of the interface was really good, with an average SUS score of $\bar{x} = 93.5$, $s = 4.12$ (the maximum score is 100). In addition, we asked the participants to evaluate the interface according to the following criteria in a 1-7 Likert scale (the higher, the better):

- Usefulness: ($\bar{x} = 5.2$, $s = 0.42$);
- Quality of the generated stories ($\bar{x} = 5.0$, $s = 0.67$);
- Correspondence between the story and the images ($\bar{x} = 5.3$, $s = 0.79$);
- Correspondence between the story and the literary genre ($\bar{x} = 5.2$, $s = 0.82$);
- Editing features ($\bar{x} = 6.7$, $s = 0.48$);
- Global interface evaluation ($\bar{x} = 5.3$, $s = 0.48$).

The results show a good appreciation of both the features and the interaction with the interface. However, the open-ended questions highlighted that the participants would like a broader set of objects recognizable in equirectangular images, that at the moment are limited to the employed dataset. To overcome this, they suggested to include the possibility to create bounding boxes on the objects that are present in the image but not recognized.

6 Conclusion and Future Work

In this paper, we presented an interface for supporting End-User Developers in creating point and click games through the automatic generation of stories from the game setting images. We generate the stories using a modular neural network consisting of an encoding phase, which analyses the objects in the images and associates its caption, and a decoder, which generates the story text applying the style corresponding to different literary genres such as adventure, science fiction, fantasy, thriller, horror. We also allow the user to use equirectangular images for the story generation, using another neural network module that detects objects

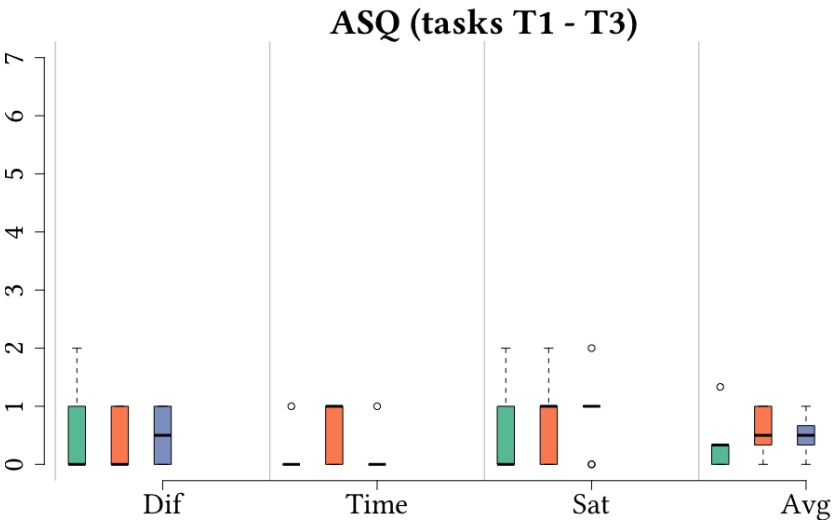

Fig. 9. After Scenario Questionnaire [16] results

in this kind of images and a process that leverages UV mapping and tangent plane projection to generate perspective images for each object detected, feeding them to the story generation network. The interface allows the user to select the input images and to control some generation parameters, represented to the end-user as the relevance of the object detected in the image and a global randomness value. Furthermore, the end-user can select which story genres will be generated and the objects she considers interesting in the equirectangular images. Besides, end-users can tailor the story modifying and saving the generated text. The user evaluation shows a good acceptance of both the features and the interface. The participant found the story generation very useful and appreciated the overall quality and the relationship with the provided input. An aspect to improve is the management of the objects detected in equirectangular images, which should be revised.

In future work, we aim at extending the recognizable objects in equirectangular images, by both annotating new object categories and allowing the user to add new bounding boxes in addition to those already recognized by the model. Also, we plan to include more user-controllable parameters, to support a more fine-grained control on the final result for the end-user and to provide more story genres.

References

1. Brooke, J.: Sus: a "quick and dirty'usability. Usability evaluation in industry p. 189 (1996)
2. Cau, F.M., Mereu, A., Spano, L.D.: Intelligent assistance for end-users in creating point and click games storylines. In: 2020 IEEE International Conference on Humanized Computing and Communication with Artificial Intelligence (HCCAI). pp. 67–73. IEEE (2020)

3. Chen, X., Fang, H., Lin, T.Y., Vedantam, R., Gupta, S., Dollár, P., Zitnick, C.L.: Microsoft coco captions: Data collection and evaluation server. arXiv preprint arXiv:1504.00325 (2015)
4. Chou, S.H., Chen, Y.C., Zeng, K.H., Hu, H.N., Fu, J., Sun, M.: Self-view grounding given a narrated 360deg video (2017)
5. Chou, S.H., Sun, C., Chang, W.Y., Hsu, W.T., Sun, M., Fu, J.: 360-indoor: Towards learning real-world objects in 360deg indoor equirectangular images (2019)
6. Deng, J., Dong, W., Socher, R., Li, L., Kai Li, Li Fei-Fei: Imagenet: A large-scale hierarchical image database. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition. pp. 248–255 (2009). https://doi.org/10.1109/CVPR.2009.5206848
7. Fanni, F.A., Senis, M., Tola, A., Murru, F., Romoli, M., Spano, L.D., Blečić, I., Trunfio, G.A.: Pac-pac: End user development of immersive point and click games. In: International Symposium on End User Development. pp. 225–229. Springer (2019)
8. Gonzalez-Rico, D., Fuentes-Pineda, G.: Contextualize, show and tell: A neural visual storyteller. arXiv preprint arXiv:1806.00738 (2018)
9. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask r-cnn (2018)
10. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition (2015)
11. Hossain, M.Z., Sohel, F., Shiratuddin, M.F., Laga, H.: A comprehensive survey of deep learning for image captioning. ACM Computing Surveys (CSUR) **51**(6), 1–36 (2019)
12. Hsu, C.C., Chen, S.M., Hsieh, M.H., Ku, L.W.: Using inter-sentence diverse beam search to reduce redundancy in visual storytelling. arXiv preprint arXiv:1805.11867 (2018)
13. Huang, T.H., Ferraro, F., Mostafazadeh, N., Misra, I., Agrawal, A., Devlin, J., Girshick, R., He, X., Kohli, P., Batra, D., et al.: Visual storytelling. In: Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. pp. 1233–1239 (2016)
14. Kiros, R.: Neural storyteller (2019), online, accessed 2020-08-17
15. Kiros, R., Zhu, Y., Salakhutdinov, R., Zemel, R.S., Torralba, A., Urtasun, R., Fidler, S.: Skip-thought vectors. arXiv preprint arXiv:1506.06726 (2015)
16. Lewis, J.R.: Psychometric evaluation of an after-scenario questionnaire for computer usability studies: the asq. ACM Sigchi Bulletin **23**(1), 78–81 (1991)
17. Li, S., Song, W., Fang, L., Chen, Y., Ghamisi, P., Benediktsson, J.A.: Deep learning for hyperspectral image classification: An overview. IEEE Transactions on Geoscience and Remote Sensing **57**(9), 6690–6709 (2019)
18. Lin, T.Y., Dollár, P., Girshick, R., He, K., Hariharan, B., Belongie, S.: Feature pyramid networks for object detection (2017)
19. Liu, Y., Fu, J., Mei, T., Chen, C.W.: Storytelling of photo stream with bidirectional multi-thread recurrent neural network. arXiv preprint arXiv:1606.00625 (2016)
20. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
21. Smilevski, M., Lalkovski, I., Madjarov, G.: Stories for images-in-sequence by using visual and narrative components. In: International Conference on Telecommunications. pp. 148–159. Springer (2018)
22. Wang, X., Chen, W., Wang, Y.F., Wang, W.Y.: No metrics are perfect: Adversarial reward learning for visual storytelling. arXiv preprint arXiv:1804.09160 (2018)
23. Yang, W., Qian, Y., Cricri, F., Fan, L., Kamarainen, J.K.: Object detection in equirectangular panorama (2018)

24. Zhu, Y., Kiros, R., Zemel, R., Salakhutdinov, R., Urtasun, R., Torralba, A., Fidler, S.: Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. arXiv preprint arXiv:1506.06724 (2015)
25. Zhu, Y., Kiros, R., Zemel, R., Salakhutdinov, R., Urtasun, R., Torralba, A., Fidler, S.: Aligning books and movies: Towards story-like visual explanations by watching movies and reading books (2015)
26. Zijlstra, F.R.H., Van Doorn, L.: The construction of a scale to measure subjective effort. Delft, Netherlands **43**, 124–139 (1985)