Coherent Stylization for Stereoscopic Augmented Reality

Max Rensen
Supervisors: Micheal Weinmann, Baran Usta, Elmar Eisemann
EEMCS, Delft University of Technology, The Netherlands

A Dissertation Submitted to EEMCS faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering

Abstract

In this paper, a method is proposed for stylizing stereoscopic augmented reality, while attempting to retain temporal and visual coherence. By tracking an AR marker and estimating a disparity map from stereo images, world positions of anchor points are tracked across frames and images, in order to move them with the scene. Additionally, the density of anchor points in the image is adjusted each frame to prevent areas from being over- or underpopulated with anchor points.

The method is applied to a simple mosaic style and both quantitative and qualitative results demonstrate improved coherence when compared to a baseline. Besides that, the algorithm runs at near-interactive frame rates.

1 Introduction

Combining the real world with the digital world has received considerable attention over the years, especially in the form of augmented reality (AR), which has made significant progress and has many applications [2, 16]. However, one of the issues facing AR is the difficulty in combining the real world with a virtual scene. These are two different media, as the real world is depicted as a 2D image or video and the virtual scene is typically a 3D rendering. As a result, virtual objects are likely to be distinguishable if they are not rendered in a photorealistic manner [1].

Another possible way to overcome this shortcoming and possibly improve user immersion, is using image/video stylization [8]. In the case of AR, this means similarly styling the real world and the virtual scene, in such a way that they blend together. There has been extensive research in stylizing images and video [11]. Additionally, it has been shown that some abstractions can improve recognition of objects by participants, both for regular video [22] and for augmented reality [19].

Some promising work has been conducted in this combined area, for example Fischer, Bartz, and Straßer apply a basic cartoon style [8] to an augmented scene, and later also a painting-like style with brush strokes [7]. This demonstrates that image stylization in AR has merit and can consistently stylize the combined real world and virtual scene. However, the aforementioned research only performs stylization on a frame by frame basis and does not take into account the preceding frames, which can result in inconsistent stylization between subsequent frames of a video for certain styles. Achieving this coherence over time, frequently called temporal coherence, has received attention for regular video stylization [3]. Additionally, a few papers address this issue in AR. For example Wang et al. use edge flow to improve frame-to-frame edge detection coherence in a cartoon style [21] and Chen, Turk, and MacIntyre use feature tracking to move brush strokes with the scene [5].

An underexplored area is the combination of coherent video stylization in AR with stereoscopic rendering of AR scenes, which means having one video stream for each eye, with the goal of better depth perception. Stereoscopic augmented reality has been shown to improve emotional involvement over just having a single video stream [18]. A paper by Lerotic et al. demonstrates a useful application of this area, by applying a non-photorealistic stereoscopic AR overlay to assist in surgery [12]. However, this is mostly unrelated to this work, as the non-photorealism is only applied to the virtual scene and dissimilar to the aforementioned image stylization. Another paper by Steptoe, Julier, and Steed performs a case study on the improvement of discernability with a simple stylization applied to stereoscopic AR [19], showing improved immersion when compared to an unstylized version, similar to the results discussed by Winnemöller, Olsen, and Gooch [22]. However, there are no significant contributions in defining a more generalized framework for coherently stylizing stereoscopic AR scenes, especially one that makes use of the extra information gained by stereo-vision, which is what this paper sets out to address.

In this paper, a new method for coherently stylizing stereoscopic AR scenes is proposed, along with the necessary restrictions on respective parts of the stylization pipeline. The coherence is not only between consecutive frames in a video, but also between the two stereoscopic frames. This new method is applied to a simple mosaic style for demonstration purposes, but with some small adjustments can likely be applied to any style consisting of anchor points. The main principle behind the algorithm is moving anchor points with their approximated world space coordinates, which are estimated from the stereo depth information and AR marker tracking. The most prominent restriction is the requirement for the entire scene to be static, with an exception for the virtual object, otherwise the world space position of an object changes when it is moved, and it cannot be tracked. Finally, the following research question is answered in this paper:

How can visual and temporal coherence be improved when stylizing stereoscopic augmented reality?

We begin this paper with a more verbose description of related work on coherence in Section 2. We then provide a breakdown of the prepared method in Section 3, followed by details on the implementation in Section 4. After that, the corresponding results are shown in Section 5 and discussed in Section 6. Finally, a short section on how the research in this paper has been conducted responsibly in Section 7, followed by a conclusion and recommendations for future work in Section 8.

2 Related Work

Since little work has been done regarding the combination of temporal and visual coherence in stereoscopic augmented reality styliztion, works addressing these issues will be compared to this paper separately.

Firstly, a method addressing temporal coherence in stylized AR is proposed by Chen, Turk, and MacIntyre, their method tracks feature points in the scene across frames and moves anchor points using barrycentric coordinates between the three nearest, co-existing feature points [5]. These anchor points define the position of certain style elements, in their case brush strokes, but can be adapted to other anchor point based styles, such as the mosaic tiles used in this paper. This relocation of anchor posed based on object features is similar to the method proposed in this paper, but their method suffers from the difficulty of tracking feature points across frames,

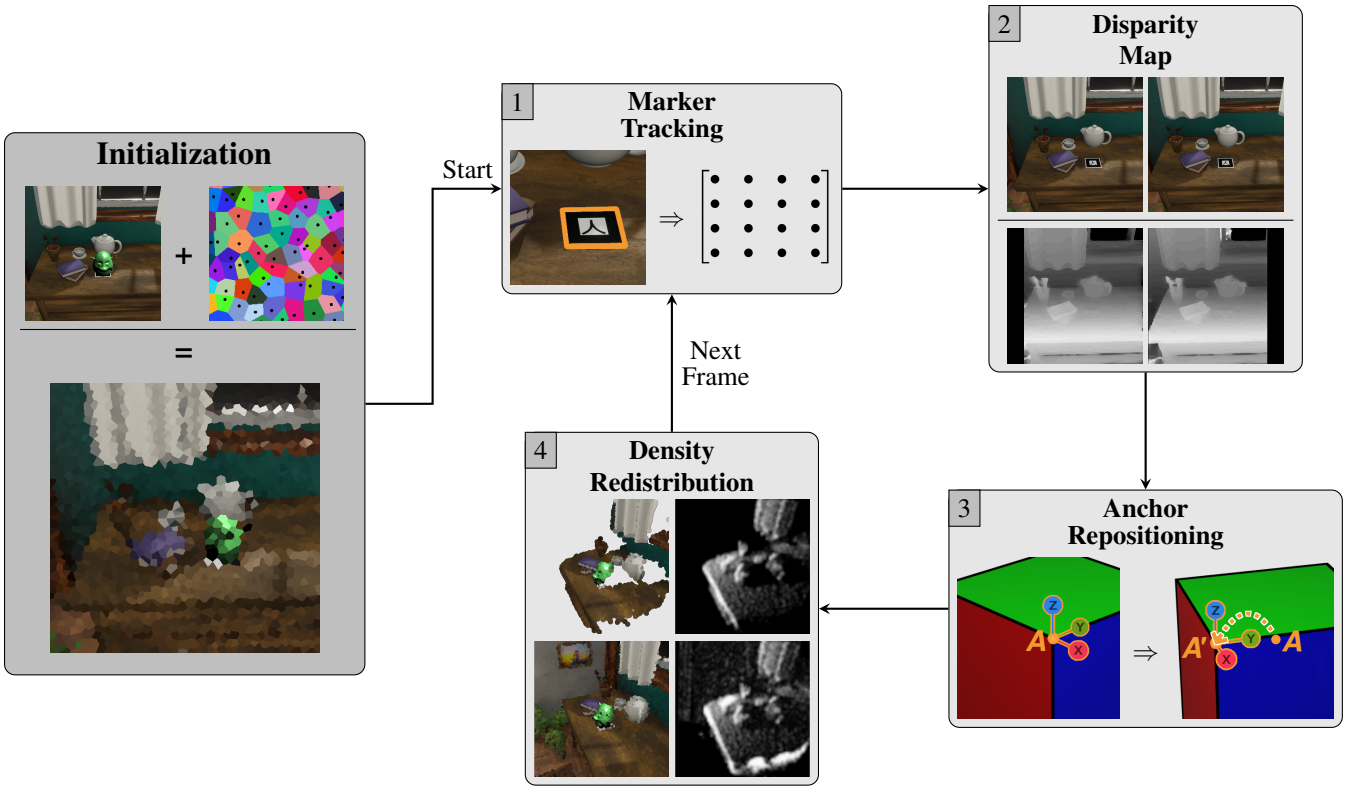


Figure 1: Overview of the coherent stylization pipeline. The initialization step sets up the initial mosaic pattern by generating anchor points and their world positions. Then a four-step process is run each frame that consists of (1) tracking the camera view matrix from the AR marker, (2) computing view space maps from disparity maps, (3) repositioning anchor points based on estimated world positions and (4) redistributing anchor density.

which we overcome by making use of estimated world space information in the form of disparity maps instead. Later, the same authors improve their brush stroke method by making use of information about the scene geometry to move anchor points according to positions in object space [4]. Again, this takes a similar approach to the one proposed in this paper, even more so as it also makes use of world space information, but it assumes the model space is given beforehand and does not currently run at interactive frame rates, partly due to the chosen brush stroke stylization. Lu, Xiao, and Tang propose a coherent video stylization technique with improved optical flow by using a machine learning based approach to track the flow of entire objects and apply affine transformations to corresponding style elements [14].

Secondly, visual coherence, or stereo coherence, is not as well-defined as temporal coherence, since it relates to stereoscopic imagery, which is far less commonplace than regular video. The work by Richardt, Kyprianidis, and Dodgson shows improved stereo coherence when applying styles in object space instead of image space, as by their user study [17], which is the same principle behind this paper.

Both coherence types suffer from a lack of generalized quantitative analysis metrics, for which a solution will not be proposed in this paper, but it will be addressed by comparison to a baseline.

3 Methodoloy

The method for coherent stylization used in this paper consists of the initial setup, followed by a four-step process executed on each frame. A respective overview of the entire method is shown in Figure 1. In the initial setup a randomized mosaic pattern is first generated for the left image, then for each mosaic anchor point, its world position is estimated, which is then used for the rest of the four-step process. Firstly, the AR marker is tracked on both stereo frames to determine the relative view information for the scene. Secondly, the stereo frames are used to compute disparity maps, from which the depth maps are estimated. Thirdly, the information from the first two steps is used to determine the expected new location of each anchor point and verify whether it is occluded or not. Finally, the density of anchor points is determined in order to add or remove anchor points in sparse or populated ares respectively. Whenever a new tile is added in the fourth step, its corresponding world position is also established. More details on the initialization and each of the steps follows below. It is important to note that the entire scene is expected to be static, except for virtual objects, otherwise the world space position of an object changes when it is moved, and any anchor point attached to it will not move with the object.

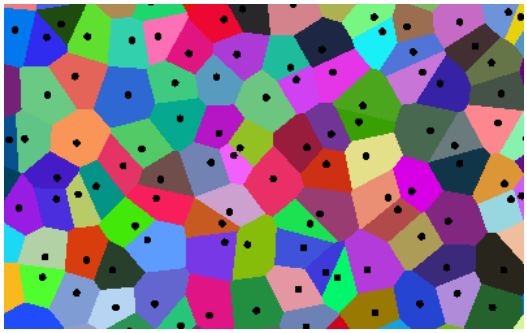


Figure 2: Example of a Voronoi diagram, where the black dots represent the points.

3.1 Initialization

In the initialization step, the initial mosaic pattern is generated for an arbitrary reference image, in this case the left stereo image. The creation of the mosaic pattern is done using a semi-randomized Voronoi diagram, which colors a pixel according to the closest point in the diagram, see Figure 2. This Voronoi diagram is generated by first uniformly distributing $M \times N$ points over an image with dimensions $W \times H$, then each point is randomly offset in the range $[0, \frac{W}{M})$ for x and $[0, \frac{H}{N})$ for y [6]. Here M and N can be chosen by the user and define the amount of horizontal and vertical anchor points in the initial mosaic pattern, W and H are simply the width and height of the input image/video. Once these points have been generated, each point is assigned the color of the closest pixel in the image, which is then used to generate the final image, as shown in Figure 3.

Figure 3: The same image of a real scene with a virtual dragon, unstylized (left), and with a 40×40 mosaic stylization (right).

The rendering of this Voronoi diagram would ideally be done by rendering infinitely long cones at each anchor point. However, this is impossible to render, so it can be approximated with finite cones made from polygons instead, which will later be exploited for generating the density in Section 3.5. The scale of these cones, meaning the radius of the circular base, should be at least large enough to cover its entire range in case each nearby anchor point is at the furthest possible position, see Figure 4.

$$radius = \sqrt{\left(\frac{W}{M}\right)^2 + \left(\frac{H}{N}\right)^2} \quad (1)$$

Finally, the anchor points are translated from 2D image space to 3D world space in order to be used in the rest of the process, this conversion is further elaborated in Section 3.6.

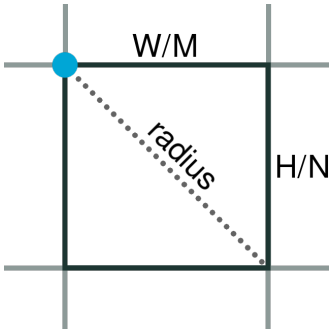


Figure 4: Minimum scale for each cone to always cover every pixel of the image.

3.2 Marker Tracking

The first step in the process, after the initial setup has been completed, is to determine a relative camera view matrix from the AR marker. An example of such a marker, which was chosen arbitrarily, is shown in Figure 5. This step is performed by ARtoolkitX and OpenCV using the scale of the AR marker and requires the marker to be fully visible at all times. Alternatively, scene features could be used to similarly estimate the camera view matrix. However, a marker is chosen since it is already commonplace in AR applications and is designed to be more reliably tracked.

Figure 5: Example of a typical marker used in AR scenes. Identified by a unique shape surrounded by a thick black border.

3.3 Disparity Map

Next, the stereo images belonging to the same frame are used to compute an approximate disparity map. These stereo images are assumed to be calibrated, such that any lens distortion is removed, and rectified, such that the images appear to be taken from parallel cameras. The disparity map contains the disparity for each pixel, which holds the distance between the same pixel in the two images; computed using the semiglobal matching algorithm [9] in OpenCV. The algorithm tries to find similar regions of pixels in both images to compute their disparity. After computing the raw disparity for both stereo images, an improved weighted least squares filter

from OpenCV is applied to both disparity maps, in order to smooth the result and attempt to fill in occluded or untextured areas [13]. An example of the conversion from two stereo images to disparity and from disparity to filtered disparity is shown in Figure 6.

Figure 6: The steps involved in converting stereo images (left), to disparity maps (middle), to filtered disparity maps (right). Only the respective versions of the left stereo image is shown (with increased contrast).

With this filtered disparity, the view space depth or Z-coordinate can be computed as follows [10, Chapter 11]:

$$Z = \frac{b * f}{d} \tag{2}$$

Where b is the baseline, or distance between the camera lenses, f is the focal length of the cameras, and d is the disparity. Additionally, the view space X- and Y-coordinates can be computed as follows:

$$X = \frac{(x + W/2) * b}{d} \tag{3}$$

$$Y = \frac{(y + H/2) * b}{d} \tag{4}$$

Where b and d are the same as before, and x and y are the horizontal and vertical pixel positions in the range $[0, W)$ and $[0, H)$ respectively for an image with dimensions $W \times H$. These coordinates might need to be negated depending on the desired up and right direction of the camera. Now each pixel in both stereo images has an estimate of its corresponding view space coordinate. The view space coordinates of the virtual model are added to this view space coordinate map by rendering them directly to a texture without any post-processing and imposing that texture on top of the map.

3.4 Anchor Repositioning

Once the view space coordinate for each pixel has been approximated, it can be used in conjunction with the computed view matrix to reposition the anchor points. The current relevant information stored for each anchor point is its world position, which is converted to view space with the previously estimated view matrix and this view space coordinate is subsequently converted to image space with a projection matrix, in order to determine whether the anchor point should still be visible, and if so, reposition it to the new image space coordinate. This projection matrix should be a perspective projection matrix based on the properties of the camera, such as the field of view and dimensions. The image space x- and y-coordinates are used to sample from the view space coordinate map and the distance between this sampled view space coordinate and the computed view space coordinate is compared to a value ϵ in order to determine whether it should be discarded or not. If the distance is larger than ϵ, the anchor point is likely occluded, so the anchor point ought to be removed. On the other hand, if the distance is smaller than or equal to ϵ, the anchor point is likely in the correct place, so its image space coordinate needs to be repositioned to the computed image space coordinate. It is important to note that the value of ϵ is highly dependent the scale of the AR marker in relation to the rest of the scene and the accuracy of the disparity map, in this paper a value of $\epsilon = 0.2$ is used.

3.5 Density Redistribution

The final step of the process is to redistribute the density of anchor points. This is necessary because removing occluded anchor points from the previous step might result in areas with few style features. In this case it would result in large areas with the same color. Additionally, when the surface of an object was initially in frontal view and is now viewed from the side, anchor points might clump together, resulting in areas that are overpopulated with anchor points and do not convey the intended style.

The density redistribution first requires the density of the anchor points to be computed for both stereo images. To compute this density map, ideally the distance to nearby anchor points is computed and aggregated for each pixel. However, this is expensive to compute, so instead the density is computed for kernels, in this case $\frac{W}{M} \times \frac{H}{N}$ pixels in size, which only check for anchor and/or feature points inside the kernel to compute the density. Here M, N, W and H are the same variables as defined in Section 3.1. In this case, a specialized method can be applied for generating the density, making use of the approximated cones used for rendering the Voronoi diagram. This method renders each of the cones in gray scale white, with a varying opacity depending on the depth of the cone from the screen, ranging from 0.5 opacity for the closest point, to 0 opacity for the furthest parts. As a result, when rendered on a dark background, areas with many mosaic tiles will appear bright, requiring the removal of anchor points, and areas with few or no mosaic tiles will appear dark, requiring the addition of anchor points. An example density image is shown in Figure 7, which would ideally not have any areas that are significantly brighter or darker than the average color.

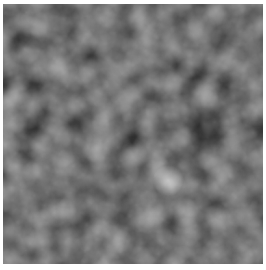


Figure 7: Example of a density image (not a density map) generated from rendering the cones from a Voronoi diagram in white with the opacity corresponding to screen depth.

The resulting image is used to compute the density ρ for each kernel by averaging the gray scale color of each pixel in the kernel. This will either indicate that an anchor point needs to be added if $\rho < \alpha$ or one or more anchor points need to be removed if $\rho > \beta$. If $\alpha \geq \rho \leq \beta$, no change is required. Here α and β can be adjusted based on the desired results, values of $\alpha = 0.01$ and $\beta = 0.9$ work sufficiently in practice and are used throughout all experiments in this paper. Adding anchor points is straightforward, the same algorithm from the initialization step in Section 3.1 is applied to generate a semi-random mosaic tile, which places the anchor point at a random position inside the kernel. Furthermore, any newly added tile has its image space coordinate converted to world space as in the initialization step, which will be elaborated in Section 3.6. Removing anchor points, on the other hand, is less straightforward since it is non-trivial to determine which anchor points to remove and which to keep. Instead, all anchor points are removed from inside the kernel and a single anchor point is added with the same procedure as before. The density redistribution process is shown in Figure 8

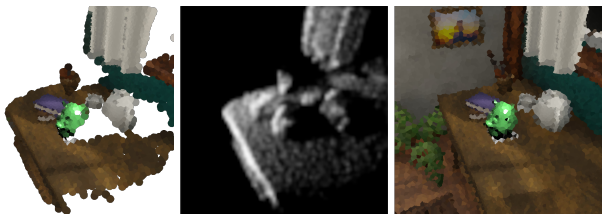


Figure 8: The steps involved in density redistribution, from an improper distribution (left), to a density map middle, to a redistributed image(right).

3.6 Image Space to World Space

To convert an image space coordinate to world space, it could be multiplied with the inverse projection matrix, followed by the inverse view matrix. However, the image space depth is not known when a new anchor point is added in image space coordinates, so the corresponding world space coordinate would be incorrect. Instead, the image space coordinate is used to sample from the view space coordinate map and convert it to world space with the inverse of the approximated view matrix from Section 3.2.

4 Implementation

Any operation performed by OpenCV is done on the CPU, which entails the marker tracking to compute the camera view matrix and the computation of the disparity map followed by the view space coordinate map. The disparity map estimation is performed on the original image size scaled down by a factor of 2. Furthermore, repositioning the anchors and redistributing the density is performed on the CPU as well, since any arbitrary amount of anchor points can be added or removed in these steps, which can be more effectively handled by the CPU. On the other hand, rendering the mosaic pattern and computing the density map is done on the GPU.

5 Results

Since a general quantitative measure for the coherence is an unsolved problem, which is non-trivial to solve, it will not be addressed in this paper. Instead, we provide a comparison to a baseline for both the quantitative and qualitative results. This baseline consists only of the initial step of the mosaic pattern generation for both the left and right image, after which all anchor points remain static in image space. A few frames, along with their stylized version, are shown in Figure 13

Pixel Differences in Numbers To measure the coherence quantitatively, pixels visible in both images are compared and their normalized difference is summed. This is done by estimating the world position of each pixel in the left image and storing it along with its normalized RGB color, using the same process described in Section 3.6. Next, just like the procedure described in Section 3.4, for each saved world position, its new image space coordinate is determined, and it is verified whether it is likely to belong to the same world position. If it does, the euclidean distance between the pixel colors are computed and summed to a total, otherwise it is ignored. This is done for both the stereo image pair of each frame and each pair of subsequent individual left and right frames. The totals are averaged for 2 scenes with 100 frames of $2 \times 1024 \times 1024$, scaled down to $2 \times 640 \times 640$, with 80×80 initial Voronoi points, the results are displayed in Table 1. For these results, the same disparity map estimation is used from the implementation, which is likely to result in imperfect view space coordinate maps. However, these imperfections will appear in both the baseline and our method, so should not bias the results.

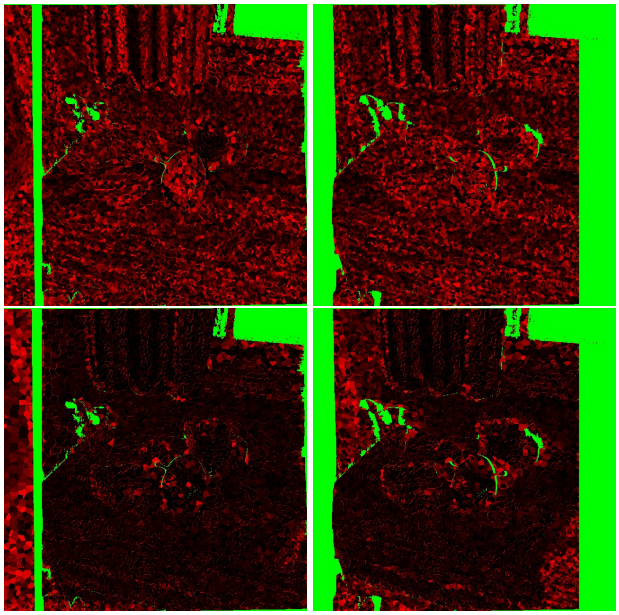
Scene	Method	Temporal	Visual
Room	Baseline	16234	16103
	Our method	9922	10991
Market	Baseline	22336	17828
	Our method	13127	15093

Table 1: Average of the total world position pixel difference between frames and images, demonstrating temporal and visual coherence.

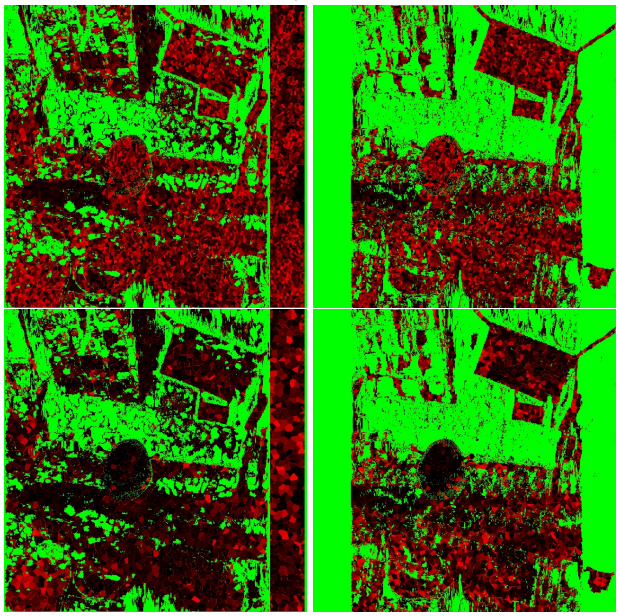
Table 1 shows that there is a large difference in pixels between the images, even for our method, which is in part due to incomplete areas in the disparity map and anchor points near disparity edges that indicate far apart objects. However, there is a statistically significant difference between the baseline and our method, namely for the room around 63.6% for the temporal coherence and 47.6% for the visual coherence, and for the market around 70.1% for the temporal coherence and 18.1% for the visual coherence.

Pixel Differences Visualized A visual example of the pixel differences is shown in Figure 9. Here, green pixels are pixels whose world coordinates are not visible in both images. For all other pixels, its red intensity indicates the euclidian color distance. To highlight the differences, each distance is normalized by dividing by a constant value η that is larger than the 95th percentile of distances, in this case $\eta = 0.1$. Large green areas are visible where no disparity information can be estimated, such as in the windows in the room scene, at the background of the market scene, and around any edges. For

the same reason, the temporal coherence images have an error distribution similar to the baseline images, since our method essentially randomizes the mosaic tiles of this side each time. The same holds for the omitted right side of the right stereo image. Besides the differences in areas with imperfect disparity, the most significant error of our method is visible around the aforementioned disparity edges.



(a) Left stereo images of the room scene.



(b) Right stereo images of the market scene.

Figure 9: Pixel differences between frames visualized for a baseline (top) and our method (bottom). Left images show temporal coherence and right images show visual coherence. Green pixels indicate pixels that are not visible in both images, the red intensity of the remaining mutually visible pixels indicates euclidean distance between the normalized RGB color, divided by $\eta = 0.1$.

Qualitative Comparison Besides a quantitative measure, it is also relevant to highlight a few qualitative results. For instance, Figure 10 shows the same books in the room scene, in the left stereo image from a different angle between two frames for both the baseline (top images) and our method (bottom images). The temporal coherence of the baseline is visibly worse than the temporal coherence of our method, especially around the areas of stark contrast, such as in the highlighted area.

Figure 10: Books stylized with a mosaic style, without temporal coherence adjustments (baseline, top) and with temporal coherence adjustments (our method, bottom).

Another qualitative example of the temporal coherence of a poster in the market scene is shown in Figure 11. Here, the right stereo image is shown between two frames for both the baseline (top images) and our method (bottom images). Again, the temporal coherence of our method is visibly better than the baseline in the highlighted area and seldom worse in surrounding areas.

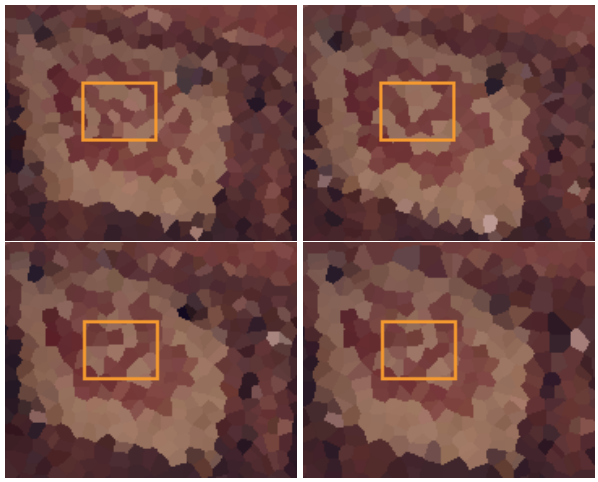


Figure 11: Poster stylized with a mosaic style, without temporal coherence adjustments (baseline, top) and with temporal coherence adjustments (our method, bottom).

Lastly, a qualitative example of the visual coherence is shown in Figure 12, where the same window frame is shown between the left and the right stereo images for both the baseline (top images) and our method (bottom images). The visual coherence of our method is visibly better than the baseline, such as in the highlighted areas.

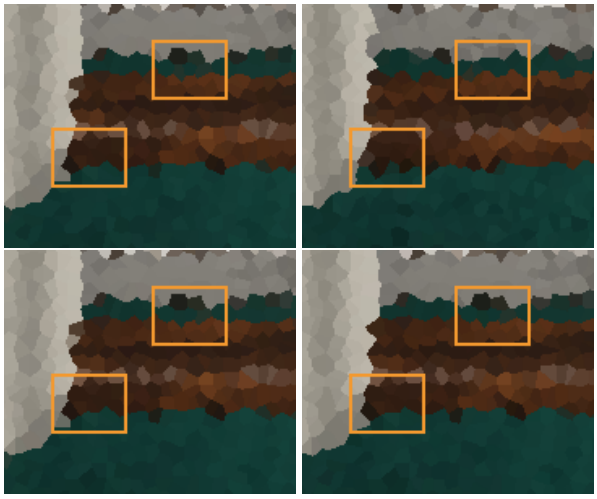


Figure 12: Window frame stylized with a mosaic style, without visual coherence adjustments (baseline, top) and with visual coherence adjustments (our method, bottom).

Frame Times Finally, a measurement of frame times and frames per second is highly relevant for an AR scenario. Table 2 shows the average FPS, corresponding frame time, alongside the frame time spent on estimating the disparity map and subsequent view space coordinates. This average is based on the same scenes and dimensions from the quantitative pixel differences. These results demonstrate 76% of the frame time is spent on this estimation, while the actual contribution of this paper relates to the remaining 24% of the process, including the marker tracking, anchor repositioning, density redistribution and of course rendering.

FPS	Frame time	Disparity estimation
5	201 ms	154 ms

Table 2: Average FPS, frame time and frame time spent on disparity estimation combined with view space coordinate calculations.

6 Discussion

The results demonstrated in the previous section make it clear that our method objectively outperforms the coherence of a baseline method, that makes no effort to achieve coherence, as by our quantitative analysis. However, the large pixel difference between frames and images in our method is still apparent from both the quantitative and qualitative results. This is especially apparent near disparity edges of objects that are far apart, since anchor points of mosaic tiles near these edges have the potential to differ significantly in color between frames, due to imperfect disparity maps. On the other hand, the coherence of objects with noticeable texture is shown to significantly improve qualitatively.

As there has been little work done in coherent image stylization for stereoscopic augmented reality, it is difficult to compare it to other works. However, the world space method by Chen, Turk, and MacIntyre [4] demonstrates similar anchor repositioning and density redistribution algorithms to those proposed in this paper. The major differences are in the mapping from image space to world space and in the fact that their method is not directly intended for stereo-vision. The mapping from image space to world space in their work makes use of information about the scene geometry, which is presumed to be available instead of estimated on demand.

Previous works have not demonstrated coherence methods for image stylization that are shown work for both temporal and visual coherence, nor make use of additional information gained from stereoscopy to achieve coherence. This work attempts to achieve this coherence for anchor point based styles, by making use of a combination of the information gained from both augmented reality and stereo-vision.

7 Responsible Research

To allow for the reproducibility of this research, any assumptions or restrictions have been described wherever necessary alongside an extensive description of the tools, methods and algorithms used in the pipeline. Moreover, even though a generalized quantitative measurement of the results is an unsolved problem, which makes it difficult to compare to other works, an attempt was made to assess the results objectively in terms of camparison to a baseline.

8 Conclusion

In this paper, we proposed a novel method for achieving temporal and visual coherence when stylizing stereoscopic augmented reality. This method makes use of the available information from tracking the AR marker and stereo-vision to track the world position of anchor points and move them across frames, while maintaining a balanced density of anchor points.

The results demonstrated in this paper show a significant improvement of both the visual and temporal coherence when compared to a baseline with no coherence adjustments. However, the main bottleneck in both performance and accuracy is the disparity map, which takes a significant portion of the frame time to generate an imperfect disparity map.

In the future, significant improvements of the algorithm could be achieved by improving the speed and quality of disparity mapping, for instance by using a more modern learning based technique [20]. On top of that, research could focus on ways of improving the coherence around disparity edges, since the proposed method lacks most in this area. Furthermore, a user study can be conducted to better asses the qualitative results and analyze possible improvements in user immersion. Another idea is to take into account differences in specular highlights between stereo images [15], which could improve both stereo matching and visual coherence. Finally, the proposed method could be adapted to different stylizations that make use of anchor points, such as the painterly brush strokes style [4].

(a) Room scene, frame 0.

(b) Room scene, frame 16.

(c) Market scene, frame 0.

(d) Market scene, frame 21.

Figure 13: A few unedited frames of two different scenes along with their stylized version, demonstrating the results of the stylization pipeline. In these videos, the green Shrek head is the virtual object.

References

[1] K. Agusanto et al. “Photorealistic rendering for augmented reality using environment illumination”. In: *The Second IEEE and ACM International Symposium on Mixed and Augmented Reality, 2003. Proceedings.* 2003, pp. 208–216. DOI: 10.1109/ISMAR.2003.1240704.

[2] Donna R. Berryman. “Augmented Reality: A Review”. In: *Medical Reference Services Quarterly* 31.2 (2012). PMID: 22559183, pp. 212–218. DOI: 10.1080/02763869.2012.670604.

[3] Pierre Bénard, Adrien Bousseau, and Joëlle Thollot. “State-of-the-Art Report on Temporal Coherence for Stylized Animations”. In: *Computer Graphics Forum* 30.8 (2011), pp. 2367–2386. DOI: 10.1111/j.1467-8659.2011.02075.x.

[4] Jiajian Chen, Greg Turk, and Blair MacIntyre. “A non-photorealistic rendering framework with temporal coherence for augmented reality”. In: *2012 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. 2012, pp. 151–160. DOI: 10.1109/ISMAR.2012.6402552.

[5] Jiajian Chen, Greg Turk, and Blair MacIntyre. “Painterly rendering with coherence for augmented reality”. In: *2011 IEEE International Symposium on VR Innovation*. 2011, pp. 103–110. DOI: 10.1109/ISVRI.2011.5759610.

[6] Jiajian Chen, Greg Turk, and Blair MacIntyre. “Watercolor Inspired Non-Photorealistic Rendering for Augmented Reality”. In: *Proceedings of the 2008 ACM Symposium on Virtual Reality Software and Technology*. VRST ’08. Bordeaux, France: Association for Computing Machinery, 2008, pp. 231–234. ISBN: 9781595939517. DOI: 10.1145/1450579.1450629.

[7] J. Fischer, D. Bartz, and W. Straßer. “Artistic Reality: Fast Brush Stroke Stylization for Augmented Reality”. In: *Proceedings of the ACM Symposium on Virtual Reality Software and Technology*. VRST ’05. Association for Computing Machinery, 2005, pp. 155–158. ISBN: 1595930981. DOI: 10.1145/1101616.1101649.

[8] J. Fischer, D. Bartz, and W. Straßer. “Stylized augmented reality for improved immersion”. In: *IEEE Proceedings. VR 2005. Virtual Reality, 2005.* 2005, pp. 195–202. DOI: 10.1109/VR.2005.1492774.

[9] Heiko Hirschmuller. “Stereo Processing by Semiglobal Matching and Mutual Information”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 30.2 (2008), pp. 328–341. DOI: 10.1109/TPAMI.2007.1166.

[10] Ramesh C. Jain, Rangachar Kasturi, and Brian G. Schunck. *Machine vision*. 1995.

[11] Jan Eric Kyprianidis et al. “State of the ”Art”: A Taxonomy of Artistic Stylization Techniques for Images and Video”. In: 19.5 (2013), pp. 866–885. DOI: 10.1109/TVCG.2012.160.

[12] Mirna Lerotic et al. “pq-space Based Non-Photorealistic Rendering for Augmented Reality”. In: *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2007*. Ed. by Nicholas Ayache, Sébastien Ourselin, and Anthony Maeder. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 102–109. ISBN: 978-3-540-75759-7.

[13] Wei Liu et al. *Semi-Global Weighted Least Squares in Image Filtering*. 2017. DOI: 10.48550/ARXIV.1705.01674.

[14] Cewu Lu, Yao Xiao, and Chi-Keung Tang. “Real-Time Video Stylization Using Object Flows”. In: *IEEE Transactions on Visualization and Computer Graphics* 24.6 (2018), pp. 2051–2063. DOI: 10.1109/TVCG.2017.2700470.

[15] Alexander A. Muryy, Roland W. Fleming, and Andrew E. Welchman. “Key characteristics of specular stereo”. In: *Journal of Vision* 14.14 (Dec. 2014), pp. 14–14. ISSN: 1534-7362. DOI: 10.1167/14.14.14.

[16] Andrew Y.C. Nee et al. “Augmented reality applications in design and manufacturing”. In: *CIRP Annals* 61.2 (2012), pp. 657–679. ISSN: 0007-8506. DOI: 10.1016/j.cirp.2012.05.010.

[17] Christian Richardt, Jan Eric Kyprianidis, and Neil Dodgson. “Stereo Coherence in Watercolour Rendering”. English. In: Symposium on Non-Photorealistic Rendering and Animation 2010, NPAR ; Conference date: 07-06-2010 Through 10-06-2010. 2010.

[18] Yoones A. Sekhavat and Hossein Zarei. “Sense of Immersion in Computer Games Using Single and Stereoscopic Augmented Reality”. In: *International Journal of Human-Computer Interaction* 34.2 (2018), pp. 187–194. DOI: 10.1080/10447318.2017.1340229.

[19] William Steptoe, Simon Julier, and Anthony Steed. “Presence and discernability in conventional and non-photorealistic immersive augmented reality”. In: *2014 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. 2014, pp. 213–218. DOI: 10.1109/ISMAR.2014.6948430.

[20] Vladimir Tankovich et al. *HITNet: Hierarchical Iterative Tile Refinement Network for Real-time Stereo Matching*. 2020. DOI: 10.48550/ARXIV.2007.12140.

[21] Shandong Wang et al. “Real-time coherent stylization for augmented reality”. In: *The Visual Computer* 26.6 (2010), pp. 445–455. DOI: 10.1007/s00371-010-0436-z.

[22] Holger Winnemöller, Sven C. Olsen, and Bruce Gooch. “Real-time video abstraction”. In: *ACM Trans. Graph.* 25 (2006), pp. 1221–1226.