Automatic Camera Control

A Dynamic Multi-Objective Perspective

Burelli, Paolo; Preuss, Mike

Published in:
Applications of Evolutionary Computation

DOI (link to publication from Publisher):
10.1007/978-3-662-45523-4_30

Publication date:
2014

Document Version
Early version, also known as pre-print

Automatic Camera Control: a Dynamic Multi-Objective Perspective

Paolo Burelli[1], Mike Preuss[2]

[1] Department of Architecture, Design and Media Technology,
Aalborg University Copenhagen, Denmark
[2] European Research Center for Information Systems (ERCIS)
Westfälische Wilhelms-Universität Münster, Germany
Email: pabu@create.aau.dk, mike.preuss@uni-muenster.de

Abstract. Automatically generating computer animations is a challenging and complex problem with applications in games and film production. In this paper, we investigate how to translate a shot list for a virtual scene into a series of virtual camera configurations — i.e automatically controlling the virtual camera. We approach this problem by modelling it as a dynamic multi-objective optimisation problem and show how this metaphor allows a much richer expressiveness than a classical single objective approach. Finally, we showcase the application of a multi-objective evolutionary algorithm to generate a shot for a sample game replay and we analyse the results.

1 Introduction

Three-dimensional computer animation is an established technique employed in the production of films, video-games, commercials and many other visual media. The idea behind 3D computer animation is to produce a virtual environment, containing elements such as lights, objects and buildings, and animate these elements while rendering the frames of a video from a specific point of view within the environments. Producing an animation, like an animated film or a game cut-scene, is a rather complex task, involving 3D modelling, animation, camera work, lighting and a number of other technical and artistic tasks. Such productions often require the work of several professionals for a span of several months. On the other hand, In the last two decades, the availability of cheap or free animation tools and the advent of customisable game engines has produced a drastic change in the demography of the virtual film makers [14]. New forms of the film medium became popular, such as machinima or game replays, created by non-professional or semi-professional cinematographers or game players.

Automating the generation of an animation or parts of it is potentially beneficial as a mean to reduce the costs of professional productions as well as to increase the quality of amateur ones. Furthermore, many of the problems connected with the automation of the cinematographic process in animations are present also in computer games — e.g. effective viewpoint animation, lighting, shot definition and selection — making this a common basic problem for most virtual reality applications. In particular, the problem of virtual camera placement and animation is a common basic virtual reality problem

which has received a lot of attention from different researchers throughout the last two decades [9]. Automatic camera control can be described as the process of finding optimal camera positions and movements in a virtual three dimensional environment given a set of requirements describing how the produced images should look like.

In the state-of-the-art of automatic camera control the problem is modelled as a real-valued optimisation problem in which the search space is defined by the possible combinations of different camera parameters (e.g. position, rotation or field of view) and the objective function is based on the level of satisfaction of a series of high-level and environment-independent requirements, such as the visibility of a particular object or the size of that object on the screen. The problem has been addressed from two perspectives: off-line generation of shots from static environments and real-time animation of camera parameters in dynamic and interactive environments.

For off-line generation of shots, the optimisation process is executed until an optimal configuration of the camera is found at each frame that needs to be generated. This approach can be used to generate static pictures; however it is not suitable for dynamic scenes either in real-time or off-line, as each shot is considered as a separated optimisation problem; therefore, subsequent optimisations might end up with very different optimal solutions and this would result in a flickering, unstable view. In the second group of approaches, the search process is performed while the virtual scene is changing — e.g. the game is played. These approaches have been successfully employed in games [4,5] and are particularly suitable for real-time tracking of optimal cameras in interactive and unpredictable virtual environments. However, due to their reactive nature, they are ill suited in situations where the optimum at a certain moment is not necessary the best solution. This is often the case in automatic camera control, in which many types of shot constraint the movement of the camera — e.g. the camera shout be fixed in one position while the characters move. In these situations the camera should be configured so that is can maximise the satisfaction of the requirements for the whole length of the shot.

The limitations of the current approaches constrain the applicability of the available automatic camera control methods; for this reason, we propose a novel approach based on multi-objective optimisation in dynamic environments. The proposed method is designed for off-line calculation of camera configuration sequences that can be used to visualise a computer animation, a game replay or to automatically place camera viewpoints during the design of a game level. Given a virtual environment, a set of events over time and a shot list — i.e. a list of shot description with starting condition and ending condition — the proposed method executes multiple multi-objective optimisations at key events or key points in time. The resulting solutions from each optimisation are subsequently used to identify the solution (or sequence of solutions) for each shot.

While the problem is dynamic, the computation of the solutions is executed off-line — i.e. after the dynamic actions is has taken place — therefore, the problem is tackled as a sequence of static problems rather than as a single dynamic optimisation problem. By addressing the camera optimisation problem from this perspective, it is possible to select solutions not only based on their instant quality, but also based on their overall performance in the shot. Furthermore, uding multi-objective optimisation, it is possible to generate different types of shots, both static and dynamic, just by changing

the principle used to select the solutions from the pool of possible solutions given by the various optimisation sequences. For instance a static shot, can be picked by finding the camera configuration which is for the longest period of time on the Pareto front during the shot.

We demonstrate the application of the approach in a 3D shooter game as a mean to automatically produce cinematographic replays of the game actions. The resulting animations are showcased and evaluated both visually and numerically against a generic single objective optimisation approach, showing that the multi-objective approach performs extremely well and shall be pursued in future works.

2 Related Work

Since the introduction of virtual reality, virtual camera control attracted the attention of a large number of researchers (refer to [9] for a comprehensive review). Early approaches focused on the mapping between the degrees of freedom (DOF) for input devices to 3D camera movement [22]. While these metaphors are currently still common in many virtual reality applications, direct manipulation of the several degrees of freedom of the camera soon demonstrated to be problematic for the user, leading researchers to investigate how to simplify camera control [12, 16].

In parallel to the research on control metaphors, a number of researchers investigated the automation of the camera configuration process. Automatic camera control identifies the process of automatically configuring the camera in a virtual environment according to a set of requirements. It is a non-linear automatic control problem [18]: the system's input are the requirements on composition and motion, while the internal state of the system is defined by the camera parameters — e.g. position and rotation. The first seminal works addressing automatic camera control according to this model [1, 13, 15] defined the concept of frame constraint and camera optimisation. These approaches require the designer to define a set of required frame properties which are then modelled either as an objective function to be maximised by the solver or as a set of constraints that the camera configuration must satisfy. These properties describe how the frame should look like in terms of object size, visibility and positioning.

Olivier et al. [15] modelled these properties — also known as frame constraints [1] — as a series of objective functions describing how much each property is satisfied by a certain camera configuration. The different objective functions are combined linearly to produce a single objective function which can be optimised either in a static environment or in a dynamic one. A variety of algorithms have been employed in the two cases including, among others, Genetic Algorithms [15, 17] and Particle Swarm Optimisation [6] for static scenes, and Hill Climbing [4] and Artificial Potential Fields [7] for real-time optimisation in dynamic scenes. The first two approaches are used to generate still images with specific composition characteristics, while the last two are designed to animate a camera in real-time interactive virtual environment — e.g. computer games.

Among other applications for automatic camera control emerged also the automatic generation of animated films or cinematographic game replays. Dominguez et al. [11] and Cheong et al. [8] investigated the process of automatically analysing a game log to generate a sequence of shot descriptions that can be used to visualise the actions

recorded in the game log. Finding the best camera configurations that fit the shot description poses a slightly different problem than the previously described ones, making classical single objective approaches not suitable. The problem is an off-line dynamic optimisation problem, that requires to find a sequence of camera configurations that optimise a given set of requirements over time in a changing virtual environment.

Using a static single-objective optimisation approach per each frame would produce sequences of camera configurations with potentially very different parameters — i.e. the camera would be jumping from one position to a potentially very different one in one frame —; moreover, there would be no possibility to control the camera's dynamic behaviour to produce, for instance, static shots as the camera will always move to the next optimal position. A local search algorithm might reduce the jerkiness of the produce animation, guaranteeing a continuity from one solution to the next one; however, such an approach would not necessarily produce the best camera configurations and it is heavily dependent on the initial configuration.

In this article, we approach this problem as a dynamic multi-objective optimisation problem. Thanks to the diversity of the solutions that can be generated trough this approach, it is possible to produce hight quality camera solutions in dynamic environment, while having full control over the dynamic behaviour of the solution. This makes possible to generate from the same set of frame constraint, different types of shots — e.g static or dolly —. In the rest of the document, we explain the details of this approach and we showcase the advantages of such approach over the current state-of-the art.

3 Multi-Objective Camera Optimisation

Even for a static optimization problem with multiple objectives, a multi-objective approach would have an advantage over a single-objective, weighted approach: depending on the shape of the Pareto front (the set of all optimal compromises for which it is not possible to further improve in one objective without deteriorating), there are situations in which any weighted approach cannot reach the front at the position that is indicated by the concrete weights, while a multi-objective algorithm can do that.

But even if we do not experience such a situation, a multi-objective approach has its advantages: its solution set is well spread over all possible weightings, such that it should contain appropriate solutions even if the problem changes slightly. A single-objective algorithm would come up with good solutions for each concrete case, but even with the same weighting, if the front changes because the problem changes, these solutions would be in a different region of the objective space, and thus most likely also somewhere else in the decision space. This would severely hamper transferability: we cannot find a good compromise solution that works well for all the points in time of a dynamic setting.

It is our basic assumption that the spread of the multi-objective result set allows for easier detection of such transferable solutions which work quite well for all time steps of a dynamic problem than the highly specialized result sets of weighted single-objective optimization runs. The results of the case study that is presented in the next section will show if this is indeed the case. Of course, the multi-objective approach also has a disadvantage because spreading a population over a Pareto front shall be more costly

than approaching just a single point on the front. However, in an offline setting as the one we treat here this disadvantage should not be too much of a problem, as there is no realtime constraint that enforces providing a solution hastily. Note that the difference between both approaches is a principle one: letting the single-objective approach run longer does not help because we may get a little bit nearer to the front, but still our best solutions would all be very similar.

In the following, we will in short explain the multi-objective algorithm we employ and then fix a criterion that enables assessing the quality of a single solution over multiple time steps, which is by nature a multi-objective measuring problem.

Our algorithm of choice is the SMS-EMOA [2], because it is known to deal well with a higher number of objectives (more than 3), and usually outperforms older approaches as the NSGA-II [10] on these settings. However, these two algorithms are conceptually not much different, we can still use the variation operators of the NSGA-II[3].

The striking differences are that a) the SMS-EMOA uses the dominated hypervolume within its selection step, and that b) only one solution is generated in every iteration and the worst is removed. This is more greedy than for the NSGA-II, but it has been shown that this leads to a false local optimum (of the multi-objective problem) only in very rare, hard to construct cases (see [3]). The hypervolume is based on the objective value differences to a fixed (bad) reference point.

Constraints can be added in a straightforward way into the algorithm, we follow the approach of a modified selection scheme as utilized in [20]: search points within infeasible regions get a penalty that resembles the distance to the next feasible region. During the selection phase, individuals that carry the highest penalties are always removed first, disregarding the quality of their other objective values. We therefore never remove a valid individual in the presence of an invalid one.

If we want to detect which of a given set of solutions is most suitable over multiple time steps and thus multiple slightly different problems, we need to define a measure and base it on a multi-objective notion. It is unlikely to find a solution that is near to specific points that are selected from different fronts by a weighting if the fronts themselves move. Thus, we relax this requirement and strive for single points that are at least very near to all fronts, regardless of which region of the front they approximate. So they are at least near optimal in some sense, even if not optimal concerning the given weighting. Our measure is related to the generational distance as defined in [21], although this was intended to assess the quality of complete populations (front approximations) and not single points. Also see [19] for a list of other frequently used measures in the multi-objective context.

At first, we need a (not necessarily optimal) reference front for every problem instance (time step) that is considered. By definition, none of the contained solutions can dominate any other.[4] For a given test point and a given front, we find all members of the reference front that dominate it or are dominated (note that a point can either dom-

[3] This refers to the *simulated binary crossover* (SBX) and *polynomial mutation* (PM) operators from [10] with (near) default parameter values of $\eta_c = 20$ and $\eta_m = 15$.

[4] A point dominates another in the objective space if it at least as good as the second point in all objectives and better in at least one.

inate or be dominated by one or multiple points of the reference front, but not both). Points that are incomparable (neither dominated nor dominate) are assigned a value of 0. If the test point dominates some points, we choose the one of them with the highest Manhattan distance (added differences per objective) and assign to it the negative value of this distance. If it is dominated, we do the same but with a positive value. Then, we iterate over the reference fronts and sum up the resulting values. This criterion is to be minimized: a value of 0 means that it is situated on all reference fronts, a negative value that it is on average better than the fronts, and a positive value means that it is on average worse than the fronts.

Note that by applying such measure, we virtually create a higher dimensional problem: if we have 5 objectives per front and 5 time steps, the resulting problem would have 25 dimensions. In principle, one could try to directly achieve a good solution by solving this 25 dimensional problem. However, present multi-objective algorithms are not at all good at working in such a setting, and it is not very likely that alternatives will emerge soon.

4 Case Study

The objective of this article is to investigate the applicability of the multi-objective optimisation to the automatic generation of cinematics. Therefore, inspired by the article by Dominguez at al. [11] on automatic generation of game replays, we employ an instance of this problem to evaluate the advantages and disadvantages of our approach. In this case-study, we have modified an existing action game called Angry Bots[5], so that we could log all the actions and position of the 3D element during a game session and we could replay these using a custom view-point. The game used in this study is an action/shooter game, in which the player controls a humanoid avatar and must explore a science fiction dungeon while been attacked by various forms for enemies. The player can move around the area and shoot at the enemies.

We recorded a short 5 seconds sequence, in which the player is approached by an enemy (both can be seen in Fig.1j). The objectives that the algorithms have to optimise correspond to the satisfaction of the following requirements for the camera:

- Full visibility for the avatar.
- Full visibility for the enemy.
- The enemy should be viewed from the back — i.e. the vantage angle should be equal to 180° horizontally and 0° vertically.
- The enemy should cover half of the screen — i.e. the projection size should be equal to 0.5.
- The enemy should be portrayed in the bottom left corner of the screen — i.e. the frame position should be equal to 0.3 both horizontally and vertically.

Each requirement has a satisfaction value that depends on the comera configuration picked as solutions. Each of these satisfaction values is defined between 0 (completely satisfied) and 1 (completely unsatisfied), and corrispond to and objective function which

[5] Unity Technologies - http://unity3d.com/gallery/demos/live-demos#angrybots

has to be minimised. For a fully detailed description of the objective function corresponding to each of these requerements, please refer to [5].

On this short game log, we have applied a multi-objective approach based on SMS-EMOA and a single-objective approach based on a standard genetic algorithm to produce a static and a follow version of the shot. In a static shot, the camera does not move for the whole shot sequence, while in a follow shot the camera moves to keep close track of the subjects on the screen.

While the GA employs elitism selection, a mutation rate of 0.5, and a crossover rate of 0.8, both algorithms (SMS-EMOA mutation/crossover parameters were given in sec. 3) run with a population of 50 individuals. The GA is allowed to run 2000 generations, summing up to 10^5 evaluations, and the SMS-EMOA runs up to 50000 evaluations but many more generations (producing only one new individual per generation). While the multi-objective approach uses the 5 objectives given above in their original form, the GA equally weights them and thus optimizes their mean value. However, for the visibility objectives, we constrained the SMS-EMOA in a way that values worse than 0.9 (0 being optimal) for these two objectives were regarded as infeasible. Not doing so would mean that we encourage the algorithm to also search for solutions were avatar or enemy or both are not visible at all, which is clearly undesired. The reference point for the SMS-EMOA was set to $(1, 1, 1, 1, 1)$, which means that the maximal hypervolume measure value is 1, and this would be attained by a single best solution that resides at $(0, 0, 0, 0, 0)$, meaning that it is optimal in all objectives.

4.1 Optimal Solution Difference Estimation

At first we are interested in seeing how different the optimal solutions over the whole sequence of 5 seconds actually are. As the front approximation obtained by means of a multi-objective approach contain much more information on a problem instance than the end population of a single-objective algorithm (as it tends to converge to a very small region of the search space), we ran the SMS-EMOA 10 times on the start time (0 seconds) and re-evaluated the final populations (which contain very different, but in the Pareto sense near optimal solutions) over the successive time steps after 1, 2, 3, and 4 seconds. The S-metric (hypervolume) measures degrade from 0.954 to 0.634, 0.299, 0.163, and 0.074. We deduce that the problem instances at the different time steps are quite different, and that good solutions for time 0 are probably quite bad for the last time steps. We can also state that the obtained final fronts usually contained between 40 and 50 individuals (50 being the theoretical maximum, the whole population is spread over the Pareto front). This means that the 5 different objectives are at least partly in conflict (not necessarily all combinations of them), and that it is highly unlikely that a solution that achieves the optimum for all 5 objectives exists.

4.2 Static and Follow Shot Comparison

We now embark on numerically and visually comparing the results of the single- and multi-objective approach for the static and follow shot. Such a comparison is not trivial, because the two approaches have very different properties concerning the provided solution set: while the MO-approach generates a whole population of different solutions

Table 1: Average objective values (ABFV) for the two approaches, static and follow shot.

shot	MO	GA best	GA average	GA avg. variance
static	0.849	1.031	1.827	0.146
follow	0.466	0.810	1.341	0.557

in every run, the GA comes up with only one and slight variations of it. Selection of a suitable solution thus is no issue for the GA, but it is for the SMS-EMOA. Thus, in addition to comparing the results of one MO run and one GA run (each repeated for the 5 time steps), we also have a look at the behavior of the single-objective approach if run 10 times (50 runs altogether), taking only the best obtained solution for each time step into account. This should on the one hand make the results more valuable from a statistical viewpoint, and on the other hand allow conclusions concerning the achievable improvement if for the single-objective approach also a larger pool of solutions exists.

The considered performance measure for both shot types is the *average best fitness value* (ABFV), which here resembles sum of the fitness values over the 5 objectives, averaged over the 5 time steps. For the static shot, only one solution is selected and the average is computed over this single solution. For the follow shot, we select the best solution returned for each time step on the GA side, and the best (in terms of averaged objectives) solution returned by the SMS-EMOA for the first time step, and the subsequently nearest solutions contained in the fronts for the next time steps.

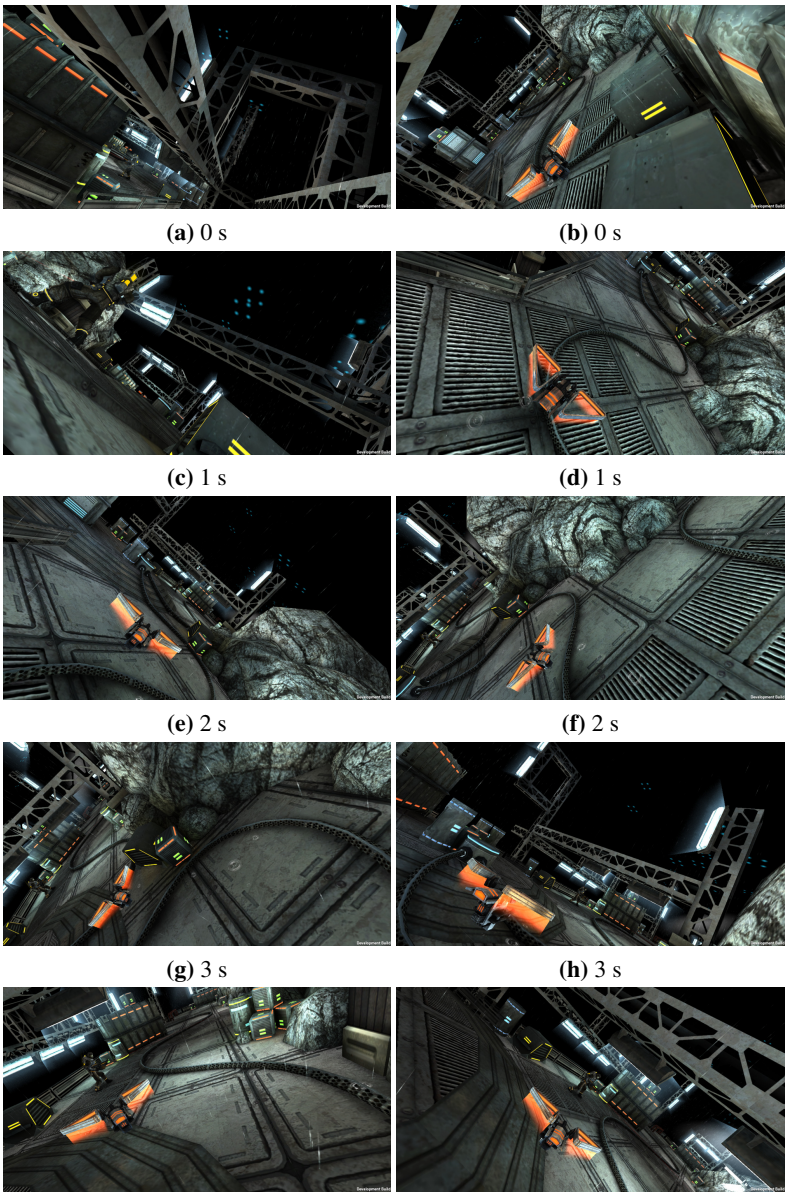
The results of the comparison are displayed in table 1, also containing the variance over the 10 GA runs as last column. The corresponding screenshots for the follow shot are provided in figure 1. We can state that the multi-objective approach leads to better results than the single-objective one on the average, and even if only the best GA solution from 10 runs is considered.

For the static shot, the visual difference is not really perceivable, it is therefore not depicted. In the follow shot, the solutions provided by the single objective are often very different, which is a big disadvantage for a video (meaning that the camera would need to make fast movements across very different solutions). This gets especially clear while looking at the first 3 frames, where the scene is shot from pretty different angles.

The multi objective approach solves this problem, because there is a large set of near-optimal but different solutions to choose from, and we can reason in objective space level. It is thus possible to find a set of solutions with minimal objective space distance, corresponding to small visual differences. In consequence, this makes the shot sequence much more smooth. Whereas the single-objective approach may lead to a video with jumpy and unsteady camera, the multi-objective approach enables a much more smooth and coherent camera positioning.

4.3 Front Approximation Distance Comparison

As a last test, we would like to know how far the single solutions generated in 10×5 GA runs and 5 SMS-EMOA runs are located from the Pareto front approximations obtained



(a) 0 s **(b)** 0 s

(c) 1 s **(d)** 1 s

(e) 2 s **(f)** 2 s

(g) 3 s **(h)** 3 s

(i) 4 s **(j)** 4 s

Fig. 1: Sequences of 5 frames representing a follow shot. Images a,c,e,g,i (left side) are produced using single-objective approach, while images b,d,f,h,j (right side) are produced with a multi-objective approach.

Table 2: Average front distance (Manhattan) and averaged distance standard deviations over the 5 time steps for the GA and SMS-EMOA approaches.

MO	MO sd	GA best	GA best sd	GA average	GA avg. sd
3.444	0.438	2.876	1.159	6.445	0.453

by the SMS-EMOA runs (as an average over the time steps). This tells us something about the stability of single solutions over the time steps, and also about the potential improvements that can be made (if we obtain solutions below the front). As measure, we employ the average Manhattan distance to the most dominating/dominated point as laid out in section 3. This comparison is again difficult to conduct fairly, as the MO algorithm provides many solutions per run and the GA only one. The overall number of solutions considered is 220 for the SMS-EMOA, and 50 for the GA.

The results are contained in table 2 and show that the GA sometimes achieves very good solutions, but mostly converges to much worse solutions than the MO approach. The best overall solution is provided by one GA run and is slightly better than the best MO generated one. As can be guessed from the high standard deviation, it is good on average because it gets below the front for at least one time step (actually, it does so for 2 time steps). However, if we review the average GA performance (averaged over each of the 10 run groups, each containing 5 runs over the different time steps), we recognize that such an event rarely happens. On average, the best GA solution is about a factor 2 farther apart from the consecutive fronts than the best MO solution (and consistently so because the average standard deviation is much lower).

We can state that in terms of average distance to the 5 consecutive time step fronts, the GA is not very stable: it may sometimes come up with very good solutions, but usually remains far from the front. However, the single very good GA solution shows that the fronts themselves are far from optimal, it should be possible to improve them, e.g., by tuning the parameters of the SMS-EMOA. This may also be done for the GA of course, but again, we deduce that the big advantage of the MO approach lies in the multitude of solutions generated per run. There seems to be a good chance that we find a suitable solution for very different applications within the delivered Pareto front approximation.

5 Conclusion

We have tested the multi-objective approach to automated camera positioning proposed in this paper for different shot types and have also conducted 2 additional tests in order to find out a) how complete Pareto front approximations generated for one time step degrade for subsequent time steps of a dynamic scene, and b) what solution quality can be expected for multiple single-objective runs in relation to the Pareto front approximations generated by our multi-objective approach.

As this it a first study in the direction of automated generation of dynamic shots for dynamic scenes, it is clear that the performance of both approaches can be improved,

e.g. by tuning or the selection of more suitable algorithms (as the CMA-ES as a most capable algorithm for the single-objective case) or search operators. However, the tackled problem is multi-objective even if only one time step is considered, and it gets only higher-dimensional (in objective space) for multiple time steps. Therefore, applying a multi-objective algorithm seems most appropriate, and indeed it shows that the high number of good but different solutions obtained from running a multi-objective algorithm pays off: it is much more likely to find suitable solutions for multiple purposes in the result set of a multi-objective algorithm than to do so even for several runs of a single-objective algorithm (which would be more costly in terms of computation time).

This work shall be extended in various ways: our implementation of different types of shots is preliminary, we ought to find better heuristics to select the most appropriate solutions, and also describe other types of shots and conduct more rigorous tests on more and different scenes. However, the key advantage of the proposed multi-objective approach is that we can reason on the fronts and in the objective space. Thus we can make informed choices when picking solutions for different shots.

References

1. William H. Bares, Scott McDermott, Christina Boudreaux, and Somying Thainimit. Virtual 3D camera composition from frame constraints. In *ACM Multimedia*, pages 177–186, Marina del Rey, California, USA, 2000. ACM Press.
2. Nicola Beume, Boris Naujoks, and Michael Emmerich. SMS-EMOA: Multiobjective selection based on dominated hypervolume. *European Journal of Operational Research*, 181(3):1653–1669, 2007.
3. Nicola Beume, Boris Naujoks, Mike Preuss, Günter Rudolph, and Tobias Wagner. Effects of 1-greedy -metric-selection on innumerably large pareto fronts. In Matthias Ehrgott, Carlos M. Fonseca, Xavier Gandibleux, Jin-Kao Hao, and Marc Sevaux, editors, *EMO*, volume 5467 of *Lecture Notes in Computer Science*, pages 21–35. Springer, 2009.
4. Owen Bourne, Abdul Sattar, and Scott Goodwin. A Constraint-Based Autonomous 3D Camera System. *Journal of Constraints*, 13(1-2):180–205, 2008.
5. Paolo Burelli. *Interactive Virtual Cinematography*. PhD thesis, IT University Of Copenhagen, 2012.
6. Paolo Burelli, Luca Di Gaspero, Andrea Ermetici, and Roberto Ranon. Virtual Camera Composition with Particle Swarm Optimization. In *International symposium on Smart Graphics*, pages 130–141. Springer, 2008.
7. Paolo Burelli and Arnav Jhala. Dynamic Artificial Potential Fields for Autonomous Camera Control. In *AAAI Conference On Artificial Intelligence In Interactive Digitale Entertainment Conference*, Palo Alto, 2009. AAAI.
8. Yun-gyung Cheong, Arnav Jhala, Byung-chull Bae, and R Michael Young. Automatically Generating Summary Visualizations from Game Logs. In *AAAI Conference On Artificial Intelligence In Interactive Digitale Entertainment*, pages 167–172, 2008.
9. Marc Christie, Patrick Olivier, and Jean-Marie Normand. Camera Control in Computer Graphics. In *Computer Graphics Forum*, volume 27, pages 2197–2218, 2008.
10. K. Deb, A. Pratap, and S. Agarwal. A fast and elitist multi-objective genetic algorithm: NSGA-II. *IEEE Trans. on Evolutionary Computation*, 6(8), 2002.
11. Mike Dominguez, R Michael Young, and Stephen Roller. Design and Evaluation of Afterthought , A System that Automatically Creates Highlight Cinematics for 3D Games. In *AAAI Conference On Artificial Intelligence In Interactive Digitale Entertainment*, 2011.

12. Steven M. Drucker and David Zeltzer. Intelligent camera control in a virtual environment. In *Graphics Interface*, pages 190–199, 1994.
13. Frank Jardillier and Eric Languénou. Screen-Space Constraints for Camera Movements: the Virtual Cameraman. *Computer Graphics Forum*, 17(3):175–186, August 1998.
14. Henry Lowood. High-performance play: The making of machinima. *Journal of Media Practice*, 7(1):25–42, 2006.
15. Patrick Olivier, Nicolas Halper, Jonathan Pickering, and Pamela Luna. Visual Composition as Optimisation. In *Artificial Intelligence and Simulation of Behaviour*, 1999.
16. Cary B. Phillips, Norman I. Badler, and John Granieri. Automatic viewing control for 3D direct manipulation. In *ACM SIGGRAPH Symposium on Interactive 3D graphics*, pages 71–74, Cambridge, Massachusetts, USA, 1992. ACM Press.
17. Jonathan Pickering. *Intelligent Camera Planning for Computer Graphics*. PhD thesis, University of York, 2002.
18. Lev Semenovich Pontriagin. *Mathematical Theory of Optimal Processes*. Interscience Publishers, 1962.
19. Ruhul Sarker and CarlosA. Coello Coello. Assessment methodologies for multiobjective evolutionary algorithms. In *Evolutionary Optimization*, volume 48 of *International Series in Operations Research & Management Science*, pages 177–195. Springer US, 2002.
20. Julian Togelius, Mike Preuss, Nicola Beume, Simon Wessing, Johan Hagelbäck, Georgios N. Yannakakis, and Corrado Grappiolo. Controllable procedural map generation via multiobjective evolution. *Genetic Programming and Evolvable Machines*, 14(2):245–277, 2013.
21. David A. Van Veldhuizen and Gary B. Lamont. Evolutionary computation and convergence to a pareto front. In John R. Koza, editor, *Late Breaking Papers at the Genetic Programming 1998 Conference*. Stanford University Bookstore, 1998.
22. Colin Ware and Steven Osborne. Exploration and virtual camera control in virtual three dimensional environments. *ACM SIGGRAPH*, 24(2):175–183, 1990.