Tackling Diverse Student Backgrounds and Goals while Teaching an Introductory Visual Computing Course at M.Sc. Level

Samuel Silva[†1]

[1]Department of Electronics, Telecomunications and Informatics (DETI) & Institute of Electronics and Informatics Engineering and Informatics of Aveiro (IEETA), University of Aveiro, Aveiro, Portugal

Abstract

Visual Computing entails a set of competences that are core for those pursuing Digital Game Development and has become a much sought competence for professionals in a wide variety of fields. In the particular case presented here, the course serves a diverse audience from Multimedia and Design students without previous knowledge in the field and low programming competences, to students that have a previous BS.c in Game Development and have already covered the basic concepts in a previous course. Additionally, the course is also offered as an elective for Computer Science M.Sc. students. This diverse set of background competences and goals motivated designing an approach to the course where each student can build on previous knowledge and have a say on its personal learning path. This article shares the overall approach, presents and discusses the outcomes, and reflects on future evolutions.

CCS Concepts

• ***Social and professional topics*** → ***Computing education;*** • ***Computing methodologies*** → ***Computer graphics;***

1. Introduction

With the appearance of a range of graduate and postgraduate programs in fields that involve direct contributions in a wide range of topics, it becomes more common that the audiences for these programs also include diverse backgrounds. The need to adapt the curriculum of courses addressing well-known topics, such as Computer Graphics and Visualization, devising an adequate approach and level of complexity to address the needs of diverse audiences, has been recognized as paramount to bring them to service in a wide range of fields [SP20, Cas21]. The development of digital games is one such field attracting students from both artistic and technical backgrounds. Additionally, the appeal of Computer Graphics to a broader audience is well known, particularly through the fascination exerted by video games, and recognized as an important competence in many graduation programs in Computer Science.

In the particular case presented here, a Visual Computing course, with its curriculum mostly centered around Computer Graphics, is offered as part of a postgrad program in Digital Game Development and as an elective to other M.Sc. programs including Computer Science and Applied Mathematics. The wide range of backgrounds entails that the devised approach should adapt to strike a balance between conveying the basic concepts to all, but in such a way that more advanced students can, nevertheless, profit from the course to further evolve their competences capitalizing, as good as possible, on their prior knowledge [FS08]. The mindset leading to how this course is structured also embraces the goal of allowing students to learn the concepts alongside developing application-level knowledge [SS19]. In this context, the goal of this paper is to share how this has been approached, for the past three years, particularly regarding how assessment is performed around a short essay and project [MGJ06], which have allowed room and flexibility to motivate students and adapt to their interests and backgrounds.

In what follows, section 2 briefly introduces the Visual Computing course, its audience, curriculum, and overall organization. Then, in section 3, it delves on the evaluation approach, detailing individual and group assignments. Following on the application of these assessment methods, for the past three editions, section 4 illustrates some outcomes of the course along with feedback obtained from students through both the institution's pedagogical questionnaires and informal conversation. Finally, section 5 reflects on the impact of the described approach and on lessons learned to move it forward.

2. Course Overview

The Master's degree in Digital Game Development is a two-year program supported on a joint effort from the departments of Computer Science and Communication and Art at *[blinded for review]*. Visual Computing is a mandatory course lectured during the first

† This work was supported by Foundation for Science and Technology (FCT) funding, in the context of the project UIDB/00127/2020.

DOI: 10.2312/eged.20241007

semester and is structured as a 13 weeks' course with a weekly class load of 3h corresponding to 6 ECTS. The following sections provide information regarding the overall context and course organization.

2.1. Intended Audience

The students enrolled in the M.Sc. in Digital Game Development present a very eclectic range of backgrounds and competences with students arriving with different graduations including Web Design and Multimedia, Game Development, and Computer Science. This is very interesting, from the program and student's perspective, since it allows that the groups assembled throughout the program to design and develop games (two one-semester long game projects during the first year) can incorporate competences covering the different dimensions of game design and development, thus providing a learning experience that encompasses a more realistic – i.e., closer to the setting found in the game industry – team setting. Additionally, and in light of increasing interest on the topic, this course is also offered as an elective for several other M.Sc. programs, with a majority of students coming from Computer Science, but also including, e.g., Electronics and Applied Mathematics. This diversity results in asymmetries regarding student competences in topics such as algebra and computer programming, key for Computer Graphics [SWLR]. In the three editions of this course adopting the approach described here, the number of students has grown from 25 to 40 per edition, with around 15-20 coming from the M.Sc. in Digital Game Development and 5-20 doing it as an elective.

2.2. Curriculum, Organization, and Instructional Materials

The course's curriculum includes the topics presented in Table 1 and covers, to different extents, four areas: Visual Perception, Computer Graphics, Image Processing, and Computer Vision. As can be observed from the time dedicated to each of these topics, the proposed curriculum places a strong emphasis on Computer Graphics and aims to convey some very introductory knowledge on Digital Image Processing and Computer Vision. Given the importance of visual perception for the topics covered during the course, the second week is completely devoted to this topic. For Computer Graphics, the curriculum follows the usual path having, as reference, the graphics pipeline; for Image Processing, it covers basic concepts regarding digital images, linear and non-linear filtering, and morphological operators; and, finally, for Computer Vision it covers basic concepts and applications of Computer Vision mostly to illustrate a machine-learning approach supported on existing models and libraries. Timely topics, such as Augmented and Virtual Reality, that can often fall within Visual Computing are addressed by an elective.

Each week, a 3-hour class is mostly divided into two: a first part where concepts are introduced and discussed; and a second part, supported on a short hands-on guide, where students explore the practical implications and implementations of the discussed concepts. The students are provided with a structured summary of the course contents as presentation slides (made available through the university's learning management system (LMS) – Moodle), also used to support lecturing, along with a list of sources to support learning, such as textbooks, and curated websites and videos. In this regard, and whenever possible, a focused list is provided (e.g., specific chapter instead of the whole book) and, if applicable, references are provided for different levels of complexity.

2.3. Hands-on Approach

The hands-on exercises should serve for the students to explore the concepts lectured in the first part of the class. Each topic has a corresponding hands-on guide along with some companion code. The guide starts slow, in the style of a tutorial, e.g., by calling attention to some detail or asking to uncomment a few lines of code, and steps-up to more complex exercises. The goal is that a less proficient student can start by being able to run the code and learn the steps required, for instance, to move geometry data to the GPU, while advanced students will be able to progress faster and implement, for instance, a multi-object scene.

At the onset of designing these hands-on and choosing the technologies to support them were a few requirements including: a) availability and multiplatform support; b) different levels of abstraction starting from mid-level; c) adequacy to different student levels; and d) easy deployment of development environment. In this context, one of the main decisions entailed that the programming language used throughout the semester should be the same, regardless of the topic: Python. This considered that, for some students, who are not well versed in programming, the need to use multiple programming languages might have a negative impact. So, students start Computer Graphics using `PyGame` and `pyOpenGL` and, at a later stage, the game engine `Panda3D`, progress into Image Processing using a Python wrapper for `OpenCV`, and experiment with Computer Vision using `mediapipe`. The choice of `Panda3D`, besides the ability to continue using Python, considered that to experiment with 3D scenes, geometric models, textures, and 3D transformations, it would be useful to use a higher level of abstraction with support to a scenegraph. Why this should be the choice, as opposed to adopting a more mainstream game engine, is left for discussion, ahead.

3. Assessment

At the onset of the definition of the course assignments, two goals were considered: (a) allow adapting to different background and interests of each student; and (b) provide continuous support and discussion of the assignments to ensure adequate complexity and exploration of relevant topics. To this end, the evaluation has two core elements: an individual short essay and a group project (see Table 2). The two assignments are subdivided in a few substages – checkpoints –, as detailed in what follows, fostering continuous assessment throughout the semester.

3.1. Assignment 1: Essay Covering Advanced Topic

The first assignment consists of writing a short essay (approx. four pages, double column) on an advanced topic in Visual Computing followed by a peer-review stage where each student reviews two of their colleagues essays.

Class	Module	1st half	2nd half	assignment checkpoint
1	I	Introduction to Visual Computing		
2		Visual Perception and Color		
3	II	Introduction to Computer Graphics	Hands-on 01	
4		2D Visualization and Transformations	Hands-on 02	choice of article topic
5		3D Visualization and Transformations	Hands-on 03	choice of project topic
6		Illuminations and Shading	Hands-on 04	
7		Geometric Modelling	Hands-on 05	
8		Texture Mapping	Hands-on 05 (finish)	in-class midterm project review
9	III	Introduction to Digital Image Processing	Hands-on 06	
10		Image Segmentation	Hands-on 06 (finish)	delivery of article
11		In-class project development		
12	IV	Introduction to Computer Vision	Hands-on 07	delivery of peer-reviews
13		Project presentation		presentation materials
exam season		Final project delivery		short-report + source code

Table 1: *Overall distribution of topics and assignment checkpoints along the semester. Each class lasts for 3 hours and is roughly divided in a first part to present concepts and a second part to explore them in a practical exercise.*

individual short essay	**45%**
essay	35%
peer-reviewing	10%

group project	**55%**
midterm review	10%
presentation & discussion	30%
final delivery	15%

Table 2: *Considered evaluation components: each student delivers an individual short essay and collaborates in a group project.*

3.1.1. Choice of Topic

An illustrative list of topics is provided for reference and for students to choose from, if they wish, but any topic can be proposed. For those proposing their topic – this is also true for those choosing from the provided list, but it is less common –, a discussion ensues, over a couple of weeks, where they are encouraged and guided to do a preliminary analysis of what they will cover and how it is important for Visual Computing. When there is a clearer idea about the focus and potential content of the article, the formal choice is registered in the LMS.

3.1.2. Peer-review

Since the articles should cover a Visual Computing topic in more depth than in class (or one that was not addressed at all), the first goal for the peer-review assignment is to expose each student to additional topics, while reviewing. Additionally, this assignment aims to contribute to develop novel competences in reviewing scientific documents, deemed important at postgrad level. To support the task, the students need to fill a form where they provide grades and comments for each essay's organization, content quality, language and clarity, and recommended and referenced sources. Each student needs to deliver the review for two essays. These are distributed based on a first random allocation followed by fine-tuning to avoid, for instance, a very technical essay being reviewed by a student with a completely opposite profile, or group colleagues reviewing their mates work. The review is double-blind and managed through the institution's LMS.

3.2. Assignment 2: Project

The goal for the project is to apply and be able to explain the basic concepts addressed in the different modules and labs. However, this is just the proposed baseline and the students are allowed to propose the topic(s) they want to address, the depth of their approach, and the context for presenting it. The different stages for choosing and developing the project assignment are designed to foster a continuous discussion between student and faculty to provide feedback on its evolution and ensure a proper level of complexity for the available time and background knowledge, and ensure the alignment with Visual Computing topics. These assignments are developed by groups of 1-3 students (2 preferred) with the level of expected work and complexity adjusted to the size of the group.

Choosing a Topic and Development Context – Students are encouraged to pursue any advanced topic that they wish to evolve, e.g., for their portfolio. The discussion about the final project is started from early on, to ensure there is time to research ideas and discuss them with faculty. Sometimes, it is at this point that students make their first approach to the platform they will use for implementation to help them decide (e.g., Godot). This stage ends with faculty approval for the project's idea.

Midterm Project Review – The midterm project review aims to ensure that students have evolved their idea and now have a concrete plan to pursue it. For this stage, students are required to prepare a very short presentation (around 5 slides advised) to pitch their idea, defend how it relates to Visual Computing, provide information on the development tools that will be used, and show evidence of any work already carried out, such as low-fidelity mockups, testing of methods, and mastery of any novel concepts or tools.

Figure 1: *Paradox Pursuit, a small game developed using PyGame and ModernGL From left to right: an overview of the created world; a demonstration of face culling; a scene were the positives effects of mipmapping and anisotropic filtering are shown; and the car navigating the streets driven by a small physics engine developed to control its movement. Credit: Rafael Gonçalves, Rafael Matos, Daniel Ferreira.*

In-class Project Development – A couple of weeks before presentation, students have one class that is devoted to project development. During this time, they can obtain feedback from the instructor on the current stage of their work – to minimize the chance of subpar projects – and get a sense of the other group's projects.

Presentation and Discussion – Each group presents its project to the whole class and performs a live demo of the work receiving feedback about positive and negative aspects. The students are encouraged to state what are their goals until the final delivery to ensure they are not too ambitious and help with prioritization of those that have the potential to have a greater impact on the quality of their work. The gap between this stage and the final delivery also allows the students time to incorporate any feedback or finalize any aspect of the work.

Final Delivery – At this point, the students deliver their project including source code, an improved version of the presentation, if applicable, and a short-report summarizing the main aspects of the work, how it relates to Visual Computing, and providing an overall account of the concepts explored. The students have the option of delivering a short written report or they can produce a short narrated video for the purpose, which is, often, the preferred choice.

4. Course Results

The approach described above has been adopted for the past three curricular years and this section provides a short overview of the outcomes with the purpose of conveying the range of topics covered in the assignments, and student feedback, both through direct feedback and the institution's pedagogic questionnaires.

4.1. Article

Table 3 lists some illustrative topics chosen by students for their essay with a balance between those initially proposed by faculty to jumpstart the choice process, and those proposed by students. As it can be observed, there is a diverse set of subjects being covered, some more technical, others more conceptual. Whenever relevant, students are nudged to cover the latest evolutions inside a topic. For instance, addressing `Nanite` when exploring level-of-detail.

Regarding the peer-reviewing stage, the delivered reviews are good, overall, and denote that the students read the articles and provide incisive comments. It is common for students to state that reviewing other's works helped them spot aspects they could improve on their work or enabled them to learn more about a new topic. Unless the review sounds rather generic (e.g., in the style of "everything is good"), students are typically awarded a good grade.

The Visual Perception of Motion and Computer Graphics
Global Illumination: Purpose and Methods
The Rendering Pipeline of the Godot Engine
Evolution of Non-Euclidean Games
The Impact of Deep Learning Super Sampling
The Novel Approaches of Luminous Engine
Selected Highlights of SIGGRAPH 2023
Realtime Pixelart Effects for 3D Scenes
The Vulkan Graphics library
Evolution and Recent Developments of GPUs
Level-of-detail for Computer Graphics

Table 3: *Illustrative topics addressed by students for the individual essay.*

4.2. Project

The topics covered by the students for the group project have been quite diverse, so far. While some students choose to cover a demonstration of the basic concepts lectured during the semester, others choose to learn and demonstrate more advanced topics. As might be expected, the topic of developing a game as grounds for exploring the Visual Computing concepts is quite common, even among students doing the course as an elective (i.e., not from the M.Sc. in Digital Game Development). Another common option is building a complex scene where the different concepts are applied.

The platform used for developing the final project has been quite diverse, ranging from more mid-level approaches using Python and an OpenGL library to a richly flavored range of game engines – sometimes learned from scratch to develop the project – including Pygame (figure 1), Panda3D, Unity (figure 2), Godot (figure 3), and Unreal (figure 5). One example of how the project lends itself to a diverse set of student skills is the work that explored Blender to demonstrate the basic concepts concerning aspects such as offline and real-time rendering, texture mapping, discrete level-of-detail, and procedural generation of content (figure 4). This was a project carried out by students that had only very basic competences in computer programming, but were experienced in using Blender for modelling and proposed to explore the 'engine' capabilities of the tool. In a few cases, students chose to interconnect both assignments covering the main concepts and state-of-the-art

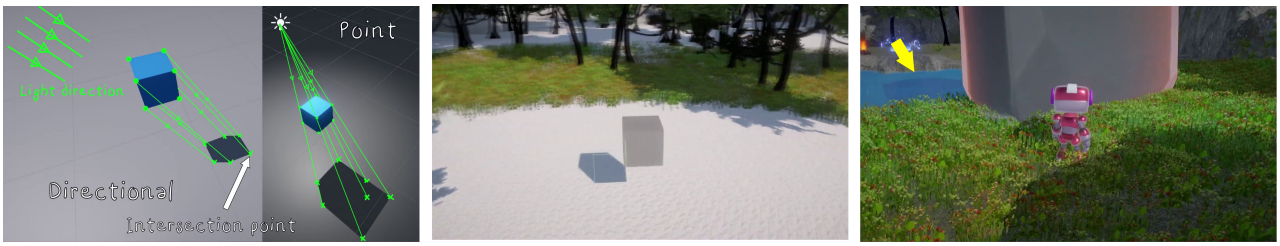

Figure 2: *In Shade, developed in Unity, students explored shadows as interactable objects. On the rightmost image, the main character needs to displace a big rock to cast a shadow over the river to be able to cross it by walking over the shadow. Credit: Daniela Silva, Gonçalo Silva, Guilherme Antunes, inspired by an original idea by PixTrick.*

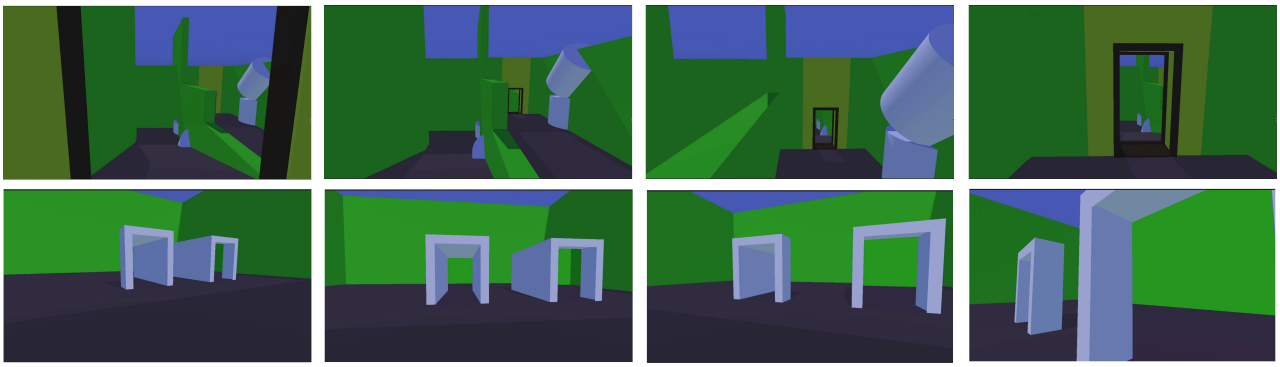

Figure 3: *Exploration of non-euclidean spaces with demos implemented using the Godot engine. At the top, implementation of the infinite room effect; at the bottom, two tunnels that have different internal and external dimensions. Credit: Daniel Ferreira, Pedro Rasinhas*

for a certain topic in the essay and, then, exploring the same topic in the project. In the latest edition, this was the case for methods to apply pixel art effects to 3D worlds, in runtime, or the exploration of non-Euclidean geometry (see Figure 3).

The evolution in quality from the in-class presentation to the final delivery of the project and short-report is also, often, quite noticeable, for many of the projects, denoting that students are committed to doing a good work. In fact, the strategy of letting students incorporate faculty feedback until final delivery is to allow students to deliver their best possible work and evolve up to the best of their capabilities within the semester's timeframe.

4.3. Course Evaluation

The approach summarized here has been in-place for the past three years and feedback has been obtained from students in two different ways: through pedagogic questionnaires and informal contact, e.g., during classes, or through contacting student representatives.

4.3.1. Pedagogic Questionnaires

By the end of each semester, our institution implements an evaluation of all courses by the students. This assessment covers a wide range of dimensions about the course and faculty. Table 4 shows a selection of the scored items based on their relevance for the discussion and covering course functioning and the perception about the faculty. The scores are presented for the two previous editions since the questionnaire for 2022-23 edition is still ongoing.

The obtained scores are very positive, overall, with the global appreciation of the course (C7) having a median of 8 and 9 for the two reported editions. The lowest score can be found for item C3, for the 2021-22 edition. This probably resulted from the fact that, in this edition, students for the MS.c in Digital Game Development were instructed to try and integrate the Visual Computing project with their work for the 2D Game Development course, running in parallel. Since the game was 2D and the Visual Computing project entails 3D, this created some pressure to find a solution that would not disrupt their game's artistic style and, hence, this may explain the lower perceived adequacy. On the subject of difficulty (C8) and workload (C9), these have scored a median of 6 or 7, indicating that the proposed approach is not perceived as overly demanding, but keeps, nevertheless, at a high level.

Finally, the devised approach for the assignments allowed to strongly motivate students to the course (F1), fostered their autonomy (F2), and created the conditions for closely monitoring their work (F4). Splitting the class in two, with an initial lecture on important concepts and a second part covering their exploration, in practical exercises, also seemed to work well (F3).

4.3.2. Informal Feedback

Besides the data obtained from the pedagogic questionnaires, students have also contributed feedback through more informal chan-

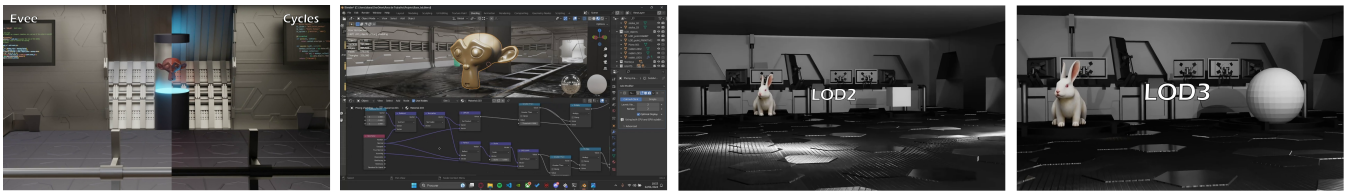

Figure 4: *A navigable lab, developed in Blender, showcasing the different concepts covered in the course. From left to right: experiments with Eevee for real-time rendering and Cycles (ray tracing), implementation of the Phong reflection model, and demonstration of discrete level of detail. Credit: Diana Silva, Leandro Ribeiro*

CHARACTERIZATION OF THE VISUAL COMPUTING COURSE		2021-22 ($N = 5$)			2022-23 ($N = 22$)		
#	**criteria**	**median**	**average**	**s.d.**	**median**	**average**	**s.d.**
C1	Coordination of the different course components	7	6.80	1.64	9	8.61	0.50
C2	Adequacy of the bibliography and study materials	7.5	7.00	2.16	9	8.75	0.55
C3	Adequacy of proposed activities	6	6.20	1.79	9	8.65	0.71
C4	Adequacy of the evaluation methods	8	7.60	1.52	9	8.65	0.67
C5	Acquired understanding about the covered topics	8	7.20	1.30	9	7.96	1.30
C6	Articulation between previous competences and proposed activities	8	7.40	1.52	8	7.00	2.27
C7	Global appreciation of course	8	7.20	1.30	9	8.57	0.79
C8	Level of difficulty of covered content	7	7.00	1.00	6	6.64	1.56
C9	Workload to obtain final approval	7	6.80	0.84	7	6.96	1.40

CHARACTERIZATION OF FACULTY		2021-22 ($N = 5$)			2022-23 ($N = 22$)		
#	**criteria**	**median**	**average**	**Sx**	**median**	**average**	**s.d.**
F1	Ability to stimulate and motivate students	8	8.00	1.22	9	8.82	0.59
F2	Incentive to student autonomy	8	8.00	0.71	9	8.43	1.38
F3	Close monitoring of student work	8	8.00	0.71	9	8.26	1.05
F4	Content and activity articulation during contact hours	8	8.40	0.55	9	8.82	0.39

Table 4: *Selected scores obtained for the Visual Computing course in the institutional pedagogic questionnaires, for the two previous editions, characterizing the course and aspects related to faculty. All items scored by students using a scale from 1 to 9. The N provided for each edition reflects the number of respondents and not the total number of students taking the course.*

nels. Since the pedagogical questionnaires above reflect feedback for the previous two editions, the feedback presented here reports to the latest (and third) edition. Regarding the curriculum, a few students sometimes voice the desire to have more advanced topics be lectured. On the opposite direction, other students consider that a high level of complexity is already in place and requires a considerable effort on their part. Regarding the hands-on activities, some students suggested that a more advanced game engine might be introduced, eventually totally replacing Panda3D or, at least, as an alternative, to allow students a choice of a more commonly used game engine. Nevertheless, they agree that some concepts might be rendered invisible by the higher abstraction level of these game engines. Regarding complexity, a student manifested the desire to have more complex activities at the end of each hands-on. Finally, concerning the assignments, the essay and peer reviewing were considered as an opportunity to learn more about a topic and about essay writing and gave them a different perspective on how it is to be the teacher when grading their work. Regarding the project, one student that already did a B.Sc. in Game Development and, therefore, had done a Computer Graphics course, stated that this course was not a waste of time for him since the freedom to choose what to explore, for the assignments, allowed evolving his knowledge on more advanced topics. Other students considered that the freedom to pursue a project they proposed strongly motivated them and allowed learning a lot more than what they expected.

5. The Road Goes Ever On

The experience in teaching Visual Computing, as described, offered first-hand information on how the devised approach works, enabling drawing some lessons, but has also shown a glimpse of where it can go to improve and provide increased learning opportunities to the students. These are discussed in the following sections.

5.1. Lessons Learned

Considering the choices made, the impact of the proposed approach, and student feedback, some reflections can be made:

Student defined projects — The considerable freedom given to the students on defining the topic of their essay and, particularly, of their project (and adopted technologies), instills a sense of student responsibility for it to work out. It should be supported on a close

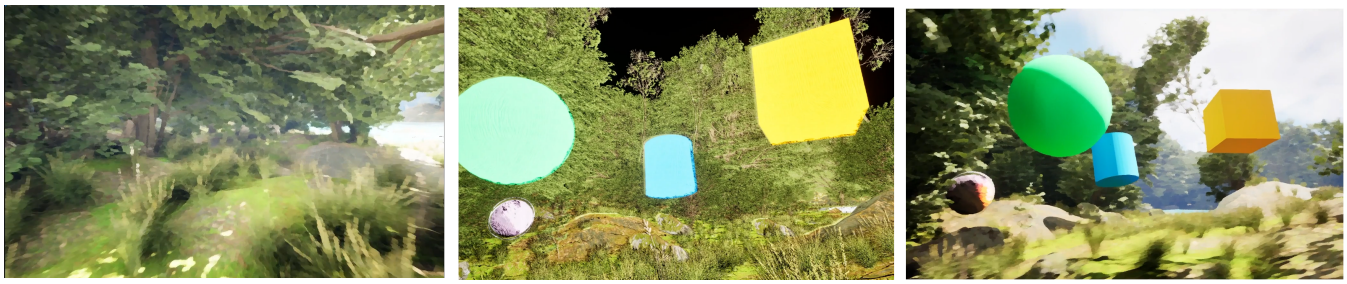

Figure 5: *Open world implemented in Unreal 5.3 exploring advanced shading including the Kuwahara filter (left), its ability to preserve edges (center) and a [car]toon shader. Credit: Gonçalo Santos, Maria Baptista, Inês Santos*

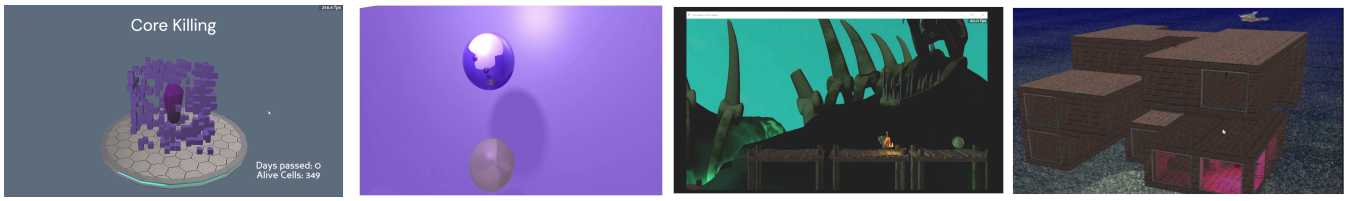

Figure 6: *Assorted examples of projects developed using PyGame (two on the left) or Panda3D (two on the right). From left to right: a 3D version of Conway's Game of Life [credit: Rúben Santo, Diogo Baptista]; a ray tracer developed from scratch [credit: Álvaro Frazão, Bóris Teixeira]; Simulation Simulator, a platformer where game progression adds features to the environment, demonstrating the different concepts (e.g., projections, shading, textures) [credit: Carolina Araújo, Francisco Pinto, Luís Chaves]; and a labyrinth – here fully shown – that can only be navigated with what the player can see using a flashlight pointed at it [credit: Lucius Vinicius, Martinho Tavares].*

monitoring from faculty to ensure that they do not go off course. In this regard, the various checkpoints, including the midterm review, along with a class specifically devoted to developing the project, allow providing feedback to minimize the chance of disaster. With a few exceptions, the projects delivered have shown quality work, student commitment, and understanding of Visual Computing concepts, providing positive support for this approach. The degree of freedom allowed on the definition of the final project was actually lower on the first edition – as alluded while discussing student feedback –, but experience showed that increasing it improved motivation and quality of the outcomes.

Not everyone loves (a) Panda — the adoption of Panda3D, while not a commonly used game engine, allows moving from the lower-level approach using OpenGL to a slightly higher level, e.g., including a scene graph, but requiring that the student keeps explicitly declaring and defining aspects such as object hierarchies, light sources, and camera positioning. This should allow them to continue grasping the different "pieces" at stake and how they relate. Additionally, it keeps students using Python, and Panda3D is often used by some of them to develop their final project (see Figure 6 for some examples), as they feel comfortable using it after the in-class hands-on activities. So, completely moving out from Panda3D does not seem warranted, for now. Nevertheless, the evolution of engines such as Godot, with a scripted language that is very similar to Python, may be an alternative to consider, in the future, also considering how easily some students learn this game engine from scratch to do the final project and how game engines have been successfully used in other contexts [Los22].

From concept to applicability and back — as previously mentioned, the course is designed for students to learn the fundamental concepts and understand their applicability in the scope of their projects. In practice, they are strongly encouraged to make it evident, when demonstrating their projects, where the different concepts are being applied and how they have mastered the proper technical language for the topic. This instills them to have these concepts in mind and strive to be able to point them out in their work with positive outcomes.

The Vulkan in the room — the Vulkan API is becoming an omnipresent topic in Computer Graphics, but this course was designed around OpenGL. In the course of the previous three editions, a few students explored Vulkan and delivered, as their essay, small tutorials on its use for basic examples. To the best of what could be judged by their feedback, the complexity behind its versatility may be a barrier for the course's intended audience and may not have a positive impact on the students' ability to understand the basic concepts. Nevertheless, on the topic of teaching Computer Graphics using Vulkan, experienced members of the community have proposed approaches that should deserve future attention [UKW23].

The end of the essay as we know it — the lesser diversity and soundness of the choices for the essay, when compared with the final project, is also something that deserves further attention, but the early stage at which it needs to be defined makes it harder for the students to have a broader vision of the field. This might be addressed with a broader set of potential topics to jumpstart the choice process. However, the essay assignment, in this time of generative large language models (e.g., chatGPT) raises some challenges on

how to implement it in a way that ensures students need to go beyond the capabilities of the language model to produce the essay. In the current edition, only a very small percentage of essays denoted this approach – along with a very small percentage of plagiarized work –, but it is bound to rise and will require action. For instance, as a starting point, in some conferences, the use of such tools is allowed during article writing, as long as there is a complete disclosure of what was the purpose, e.g., text revision. Naturally, this is an intense and open debate for the community and future editions of the course should consider and profit from it. Regarding peer-review, the students take it seriously, overall, and seems to have a positive impact on how they manage to judge their own work.

Scalability of the approach — when it comes to scalability of this approach, all project checkpoints, except for the final delivery, occur during class. So, it is fairly scalable, without being too hard on faculty, as long as classes do not move much beyond 20 students. Regarding the essay assignment, and peer reviewing, these impose a stronger effort for grading them. Nevertheless, the peer reviews can work as a first indication of essay quality that is then refined by the teacher. In fact, students are informed that their reviews will also serve this purpose. The grade they attribute to the essays, on average, is approx. 1 out of 20 points above faculty grading.

Finally, some of the positive outcomes observed, particularly regarding student autonomy and ability to define and develop the projects, may stem from the fact that all students already have a B.Sc. and, hence, an increased level of experience and maturity to tackle these matters. Therefore, the experience shared here may not be as easily replicable for undergrads.

5.2. Future Directions

As is the case for the constant evolution of the field of Visual Computing, a course tackling this topic needs to keep moving forward. As it stands, the course presented here can evolve in a number of ways, not limited to, but including:

Improve the integration among course modules — the hands-on activities covering an introduction to Digital Image Processing are still not as fully integrated with the Computer Graphics part as they could be, since students just shift to solely exploring OpenCV for that purpose, abruptly "abandoning" the Computer Graphics realm. In this regard, the lab activities can evolve to explicitly bring this synergy forward integrating Digital Image Processing, e.g., with texture mapping.

Increase the exposure to shaders — shaders have a prominent role in Computer Graphics and there is now a large amount of resources on how to develop a wide range of custom shaders. In fact, in many of the final projects, students learn and implement shaders for a variety of effects. While the current approach of the course touches on shaders, particularly regarding illumination, it should evolve to allow a more diverse exposure of students to the topic during the hands-on activities. Since many shaders delve with basic digital image processing concepts (e.g., the Kuwahara filter, Figure 5) this can be an opportunity for both deepening this topic and strengthening the integration among the Computer Graphics and Digital Image Processing modules.

Bring other game engine(s) into play — while this may not be for all students, the fact is that many seek to learn a game engine – other than Panda3D or PyGame – to develop the final project. The ideas put forward, above, may be operationalized by also introducing the option to use a game engine, such as Godot, to do it, during the final weeks of the semester, supported on new hands-on guides. While Godot may not yet be the "go-to" engine for the industry, it has been easy to approach by students and exposes them to how these tools work.

Deepen the approach towards student-centered learning — the approach in place for this course is balanced – albeit feebly and, yet, incompletely – towards a personalization of each student's learning path by allowing a choice regarding the topics and tools covered for the assignments. In this regard, it can profit from the experiences and practices from student-centered learning approaches [FS08].

The proposed approach, and the experience shared here, while based on tangible outcomes of three editions of the course, still has plenty of room for progression, in this ever evolving field. It is, nevertheless, shared in hopes that it may inspire novel and improved approaches and constructive feedback from the community.

Acknowledgements

The author thanks all the students that have contributed their commitment, projects, feedback, and suggestions along the past three years.

References

[Cas21] CASAS D.: Teaching 3D Computer Animation to Non-programming Experts. In *Eurographics 2021 - Education Papers* (2021), Sousa Santos B., Domik G., (Eds.), The Eurographics Association. doi:10.2312/eged.20211004. 1

[FS08] FROYD J., SIMPSON N.: Student-centered learning addressing faculty questions about student centered learning. In *Course, curriculum, labor, and improvement conference, Washington DC* (2008), vol. 30, pp. 1–11. 1, 8

[Los22] LOSCOS C.: Introduction to Computer Graphics: A Visual Interactive Approach. In *Eurographics 2022 - Education Papers* (2022), Bourdin J.-J., Paquette E., (Eds.), The Eurographics Association. doi:10.2312/eged.20221039. 7

[MGJ06] MARTÍ E., GIL D., JULIÀ C.: A pbl experience in the teaching of computer graphics. In *Computer Graphics Forum* (2006), vol. 25, Wiley Online Library, pp. 95–103. 1

[SP20] SANTOS B. S., PERER A.: Visualization for data scientists: How specific is it? In *Eurographics 2020 - Education Papers* (2020), Romero M., Sousa Santos B., (Eds.), The Eurographics Association. ISSN: 1017-4656. doi:10.2312/eged.20201033. 1

[SS19] SMITH G., SUNG K.: Teaching Computer Graphics Based on a Commercial Product. In *Eurographics 2019 - Education Papers* (2019), Tarini M., Galin E., (Eds.), The Eurographics Association. doi:10.2312/eged.20191031. 1

[SWLR] SUSELO T., WÜNSCHE B. C., LUXTON-REILLY A.: Technologies and tools to support teaching and learning computer graphics: A literature review. In *Proceedings of the Twenty-First Australasian Computing Education Conference*, ACE '19, Association for Computing Machinery, pp. 96–105. doi:10.1145/3286960.3286972. 2

[UKW23] UNTERGUGGENBERGER J., KERBL B., WIMMER M.: Vulkan all the way: Transitioning to a modern low-level graphics api in academia. *Computers & Graphics 111* (2023), 155–165. doi:https://doi.org/10.1016/j.cag.2023.02.001. 7