2023 HAWAII UNIVERSITY INTERNATIONAL CONFERENCES
SCIENCE, TECHNOLOGY & ENGINEERING, ARTS, MATHEMATICS & EDUCATION JUNE 7 - 9, 2023
PRINCE WAIKIKI RESORT, HONOLULU, HAWAII

A PRACTICAL EXAMPLE OF PROGRAMMING EDUCATION USING A 3DCG DEVELOPMENT TOOL WITH A PHYSICS ENGINE

MIYAMOTO, YUKINOBU
FACULTY OF BUSINESS ADMINISTRATION
KOBE GAKUIN UNIVERSITY
KOBE, HYOGO,
JAPAN

Prof. Yukinobu Miyamoto
Faculty of Business Administration
Kobe Gakuin University
Kobe, Hyogo
JAPAN

A Practical Example of Programming Education Using a 3DCG Development Tool with a Physics Engine

Synopsis:

In this paper, we describe an approach to get students interested in programming by having them learn 3DCG development at the same time as programming. Unity, a 3DCG development application with a physics engine, was introduced as the tool used, giving priority to the visual interest and the richness of the CG object library. Although most of the students had little programming experience, they were able to work on the project with great interest until its completion.

A Practical Example of Programming Education Using a 3DCG Development Tool with a Physics Engine

Yukinobu MIYAMOTO
Faculty of Business Administration
Kobe Gakuin University

Abstract

In recent years, ICT has been accelerating in educational environments, especially in the three years under COVID-19. In Japan, data science education is now being required as a common subject not only for engineering students but also for students in the humanities and social sciences. On the other hand, these data analyses involve a considerable amount of programming elements, but students who do not have a background in science or engineering still have a strong sense of dislike for this field.

In this paper, we describe an approach to get students interested in programming by having them learn 3DCG development at the same time as programming. Unity, a 3DCG development application with a physics engine, was introduced as the tool used, giving priority to the visual interest and the richness of the CG object library. Although most of the students had little programming experience, they were able to work on the project with great interest until its completion. Finally, we summarize the results of the practice at several universities and discuss the benefits of this method.

Keywords: Programming Education, Physics Engine, 3DCG Development, C#, Unity

1. Introduction

In recent years, Information Communication Technologies (ICT) has been accelerating in educational environments, especially in these three years under COVID-19. In Japan, data science education is now being required as a common subject not only for engineering students but also for students in the humanities and social sciences. Under these circumstances, Japan's Ministry of Education, Culture, Sports, Science and Technology (MEXT) is promoting education in mathematics, data science, and AI [1], as well as encouraging private universities to convert their humanities departments into science departments [2].

On the other hand, these data analyses involve programming elements, but students in the humanities and social sciences who are not in the science and engineering fields still

have a strong dislike of programming, and the challenge for universities is how to get them interested in the programming field. At the same time, students themselves wish to become human resources who can meet the demands of the times by using ICT as they wish, and there is a high demand for the establishment of ICT-related subjects. Although we are now entering an era in which students can create applications using various services without coding, it is still necessary to learn at least fundamental programming skills at the same time. We have already reported on a class practice of mobile application development using JavaScript for a learning environment that simultaneously satisfies these demands [3].

In this paper, we describe an approach to get students interested in programming and learning 3DCG production at the same time. Unity, a 3DCG production application with a physics engine, was introduced as the tool of choice, giving priority to the visual interest and the richness of the CG object library. Although most of the students in the course had little programming experience, they were all interested in the work and were able to work on it until its completion. Finally, we summarize the results of our method in several universities and discuss the benefits of our method.

2. 3DCG Development and C# Programming with Unity

This chapter describes Unity, the tool used in the proposed method. Unity[4] is a game engine with an integrated development environment (IDE) developed by Unity Technologies, Inc. A game engine here refers to a mechanism that provides significant support for game development by incorporating functions that can reproduce the laws of physics, such as gravity, friction, and bounciness, in advance. In general, it is sometimes called a physics engine for applications not limited to game development.

C# is used as a programming language that can be implemented. It also has a rich set of graphics-related functions, supporting both 2D and 3DCG. The operating platforms are IA-32, x86-64, and ARM, and it can run on both Windows and Mac PC environments. Cross-platform support is also abundant, enabling development not only for PCs, but also for mobile phones, web browsers, and VR/AR/MR devices including home game consoles.

Unity can be installed for free or for a fee. For free use, choose Unity Personal, which is available for noncommercial use and educational purposes under certain conditions. For paid users, there are three levels of functionality: Unity Plus, Unity Pro, and Unity Enterprise, respectively. Of course, the functionality is enhanced according to the amount of money paid, but since our project is within the scope of a university lecture, we chose Unity Personal, which is free of charge, due to time and financial constraints. Students

can also install Unity Personal on their own PCs to create a development environment at home, etc., as long as it is within the scope of non-commercial use. Table 1 summarizes the settings for each license type.

Table 1: Settings for each license type

Type	Fees	Concurrent Developers	Optional Functions
Unity Personal	Free	20	N / A
Unity Plus	$360 / yr.	50	Learn Premium, Cloud Diagnostics, etc.
Unity Pro	$2,000 / yr.	100	Success Advisor, Premium Support, etc.
Unity Enterprise	Individual Consultation (for Companies)		

Unity also has an external library called Assets. These assets include a wealth of digital resources such as 3D models, textures, materials, music, sound effects, and scripting packages, which can be combined like a puzzle to easily create a prototype of the work. Some free assets are often provided without warranty, and some may not be usable after a certain period of time or may not work with certain versions. Figure 1 shows an example of Starter Assets selected in the Asset Store.

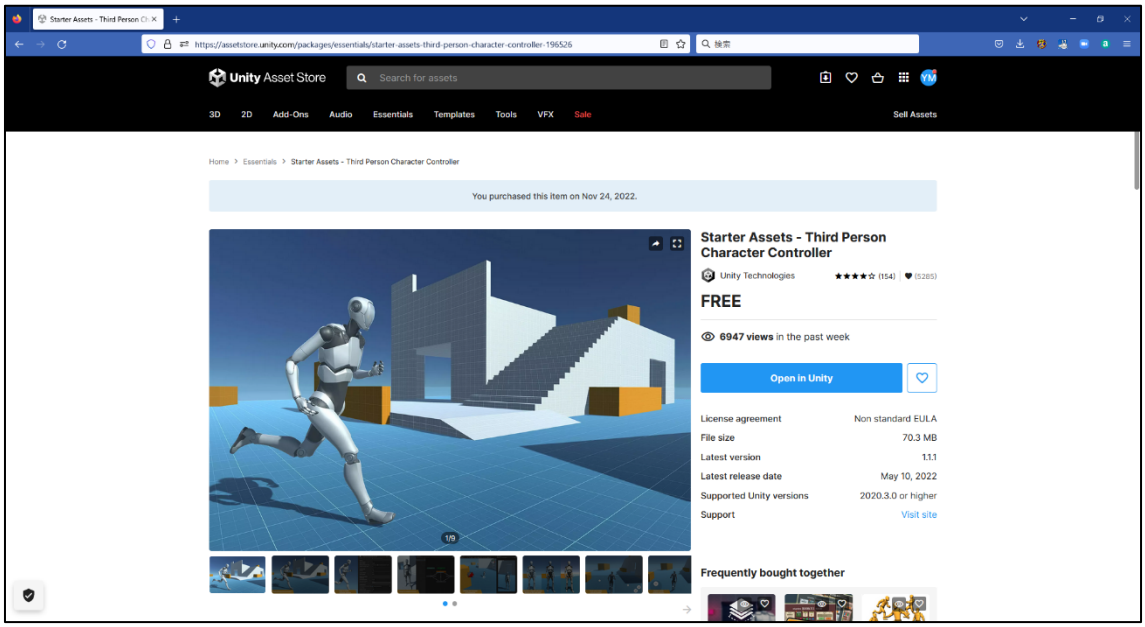

Figure 1: Example of Assets (Starter Assets)

Figure 2 shows a sample Unity development screen. As with many IDEs, the screen structure is such that the file manager, preview screen, configuration screen, message

window, and other screens can be distinguished at a glance. The editor is linked to Microsoft Visual Studio as standard because C# is used as the development language, but you can also choose a text editor to suit your personal preferences. If no editors are installed, standard Notepad can also be used.

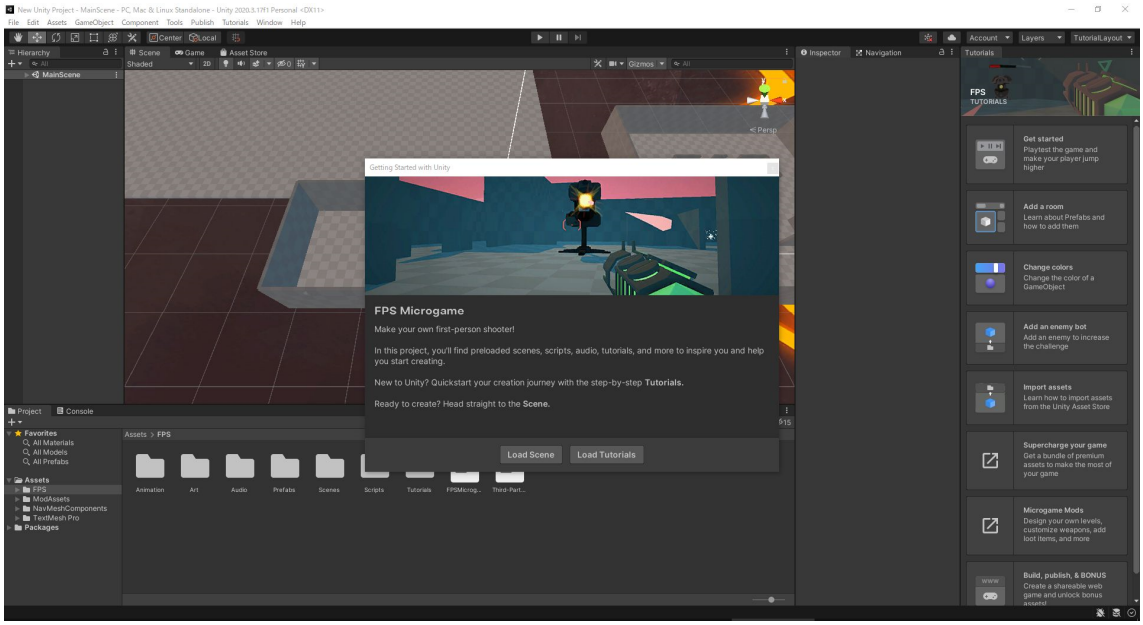

Figure 2: Example development screenshot of Unity

Unity's greatest feature is its physics engine. The physics engine is a function that incorporates the laws of physics in the natural world in advance, making it easy to create CG with motion simply by associating the laws with objects on the screen. For example, if gravity is applied to a sphere on the screen, the sphere will fall vertically if there is nothing under it and roll on a sloped floor. If friction is set between the sphere and the floor, the smoothness of the rolling motion can be varied. The bounciness coefficient can be set to adjust the way the spheres bounce when they collide. As another example, the main character of a game can run around the field, jump, and dive underwater. The ability to create not only static CG but also dynamic movies that represent objects in motion is one of Unity's great strengths. Figure 3 shows an example of configurations friction and bounciness coefficients for a spherical object. Using these configurations reduces the need to create programs for movements involving physical phenomena.

The operating environment described above enables CG and programming practice, but the following points should be considered when using Unity. First, the size of the created files is quite large. This of course affects the storage capacity of the PC, but more importantly, it takes a considerable amount of time to read and write the files to external

storage devices. In particular, when a student wants to save a work in progress at the university and take it home, it may take several minutes to copy it to an external storage device such as a USB flash memory, or even several dozen minutes in some cases. If copying is started near the end of class time, the copying may not be completed by the end of class. If the data is not stored on an external storage device but on a network drive such as the cloud, there is no need to carry the data with you, but this requires more time because the data is read/written via the network. In addition, students cannot start working simultaneously at the beginning of class unless the files in the process of being created at home are copied to the university PC in advance. These problems can be avoided if sufficient time is allocated for copying in class management and advance instruction, but it should be noted that this will shorten the actual class time.

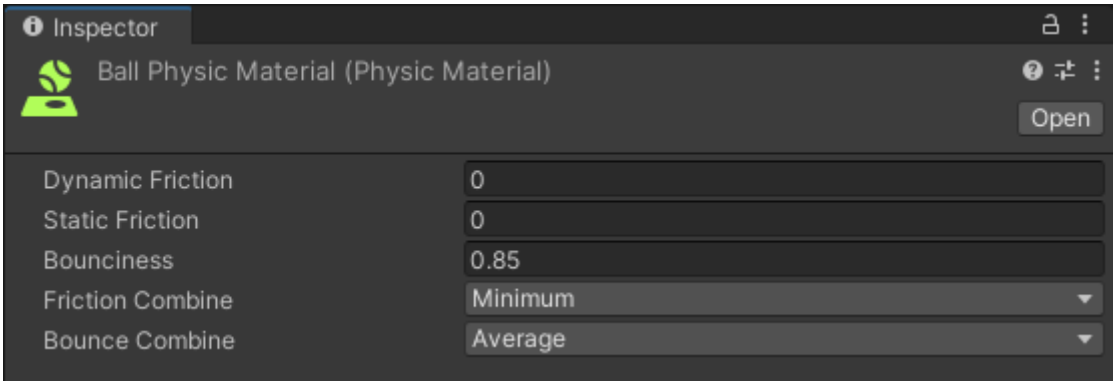

Figure 3: Example of configuration for friction and bounciness

Next, there is the issue of licensing: Unity's license is basically tied to the PC, not the user. This is not a problem if the PC is owned by an individual, since the user is always identified. On the other hand, in a free-address environment such as a university laboratory, the same PC may not be used every time. In such cases, it is possible to carry license files corresponding to individual PCs in an external storage device and load the corresponding license file when Unity starts up. In particular, recent universities are often configured to initialize to the standard state when a PC is rebooted, making it essential to carry a license file in such cases.

It should also be noted that version compatibility is not always high. The official Unity website maintains a long list of previously released versions, from which the appropriate version should be selected and installed. Intuitively, it seems that the latest version should be used, but the latest version may contain bugs that have not yet been addressed, and in some cases, the operation may be unstable. In such cases, it should be noted that the user must reject the update to the latest version when prompted to do so at startup. When using

educational materials such as textbooks, it is necessary to carefully check which version the educational materials are based on. In addition, the versions of the aforementioned Assets that work differ from one another, so there may be cases where the desired Assets cannot be installed or only some of the functions can be used. Moreover, when students install Unity on their own PCs, they need to be reminded that the versions of Unity must be the same as those used at the university. Although these version-related problems can be avoided to some extent through class management and advance guidance, it is necessary to carefully check in advance the version to be installed on the university PC and the operating environment of the sample programs in the teaching materials.

3. Examples of Class Practice

In this chapter, we describe a practical example of introducing Unity into a university class. In this project, we implemented Unity at two universities, and we will refer to University A as my main school and University B as the part-time school. At University A, the course name was “Application of Information Processing,” the implementation period was the fall semester of the 2020-2021 academic year, and the Unity Personal version 2020.3.18f was used. The number of students in each year was 6 and 12, respectively. On the other hand, at University B, the course name was “Game Programming,” the implementation period was the fall semester of the 2020 academic year, and the Unity Plus version 2019.1.4f1 was used, with 26 students in the class. Table 2 summarizes these situations.

Table 2: Lecture status for each university

University	Unity Type	Unity Version	Lecture Title	Semester	#Students
University A	Unity Personal	2020.3.18.f1	Application of Information Processing	2021 Fall	6
				2022 Fall	12
University B	Unity Plus	2019.1.4f1	Game Programming	2021 Fall	26

Next, as for the teaching materials, we adopted the reference [5] for the textbook. In this textbook, sample programs were produced based on Unity Personal version 2018.3.0f2. As we found out later, there were some cases in which the sample programs in the textbook did not work well with the Unity version of the university where the project was implemented. In addition, some of the Assets in the textbooks had already expired, and we had to replace some of them. To solve this problem, sample programs were cited from

the Website [6] as additional teaching materials, and changes were made to the course content.

Table 3 shows the class progression schedule taken from the syllabus. Each semester consists of 15 x 90-minute classes. In each class, students work on four practical tasks, and their grades are evaluated by two practical tests. Excluding the lecture guidance and review sessions, the students will only be able to work on the assignments 9 to 10 times. Each assignment is basically designed to follow the chapters in the textbook.

Table 3: Lesson schedule of each university

#Lesson	Contents	#Lesson	Contents
01st	Lecture Guidance	09th	Development practice (3)
02nd	Development practice (1)	10th	Development practice (3)
03rd	Development practice (1)	11th	Development practice (4)
04th	Fundamentals of Scripts	12th	Development practice (4)
05th	Development practice (2)	13th	Review of the second half
06th	Development practice (2)	14th	Practical test (2)
07th	Review of the first half	15th	Review of fall semester
08th	Practical test (1)	exam	N/A

4. Results and Effects of the Practice

In this chapter, we describe the practical results and effects of introducing the proposed method. Figure 4 shows some examples of a student works. The work on the left side of Figure 4 looks like a scene from a car racing game, and the work on the right side of the figure represents a fantasy world like a role-playing game.

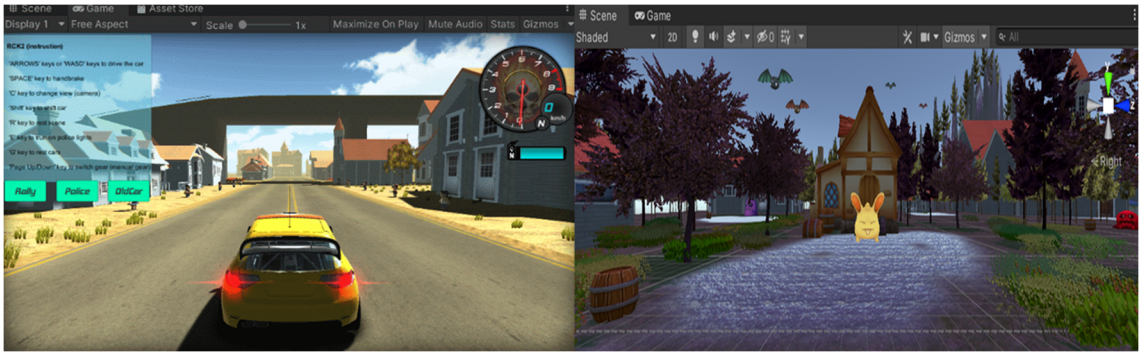

Figure 4: Example of student works

Although the students were given approximately two weeks to complete their assignments, which was not enough time due to the class schedule, we think all of the works can be highly evaluated for their full use of Unity's functionality. The key to this is the use of Assets, which contain many pre-defined parts such as cars, buildings, characters, etc., and it is possible to create a good-looking work simply by arranging these parts on a canvas. However, the overall layout and design of the project was largely dependent on the individual's sense of style, and Unity was a very useful tool for the realization of each students' ideas.

A questionnaire was administered at the end of the lecture. The questionnaire items were as follows, and the results are shown in Table 4. The results for University A are the total for the two years.

Question: How do you think you have gained knowledge, skills, and abilities by taking this lecture?

(1) Very much so
(2) Somewhat agree
(3) Cannot say either way
(4) Not really
(5) Not at all

Table 4: Responses to the question

University	#Students	(1)	(2)	(3)	(4)	(5)	N/A	Total
University A	18	**9**	**5**	0	0	0	4	18
University B	26	**5**	**13**	1	0	0	7	26

Table 4 shows that 100% of the students at University A and about 95% of those at University B responded positively to the questionnaire, indicating that students were generally highly satisfied with the program. In University A, in particular, the highest evaluation (1) was given by 2 students in 2021 and 7 students in 2022, accounting for 40% and 78% of all respondents, respectively. The evaluation of this course was higher in the second year than in the first year, which can be interpreted as a sign that the class management has become more proficient.

The questionnaire also included a free-response column at the end of the above questionnaire. The following are examples of some of the responses from University A in 2022.

- I was not very good at using computers, but by using Unity, I learned not only about CG but also how to deal with computer crashes and errors.
- I was not sure if I would be able to keep up with the class when I took the first lesson, but when I was able to make my work in the class and the 3DCG work for the final project, I think I have acquired the skills to handle a little programming.
- I didn't realize how hard it was to make a game. I learned that it is not easy to fix bugs and errors. The 3DCG production in the class was difficult but interesting.
- I am not good at expressing what I think in my head in words or sentences, so it was very fun for me to be able to express myself in the form of CG and artwork.
- I have gained a lot of respect for people who create something in these six months.
- I have gained new knowledge because I have never worked on this subject before.

The above descriptions indicate that the students were generally satisfied with the content of the classes. They also mention the difficulties they faced in creating their works and the joy they felt when they were able to achieve their goals. The students also showed respect for the creators of commercially available games. We can conclude that the introduction of this class was a great success because the students were able to confirm that they learned through actual experience the hardships, the sense of accomplishment, and the respect they received.

5. Conclusions

This paper described a practical example of 3DCG production and C# programming for social science students, and its effectiveness. The students' evaluations were generally favorable, and they were able to eliminate their dislike of programming, and at the same time, they gained a sense of satisfaction from being able to express themselves through 3DCG. The class practice described in this paper, which aimed at acquiring CG production and programming skills using Unity, can be said to have been successful enough.

The challenge for the future is to further expand the number of students. Unity has both paid and free subscriptions, with paid subscriptions requiring a license fee equal to the number of PCs running the software. For non-profit and educational purposes, Unity can be used free of charge under certain conditions, but the number of simultaneous users and some functions are limited. In the case of University A, the number of students was relatively small, so the free version was sufficient. On the other hand, University B has faculties of media, art, and architecture, and there are many contents of study that deal with CG more than usual, so the paid version was introduced and used effectively.

Now we believe that the free version is currently sufficient to achieve the educational objectives of University A. However, if the number of students who wish to take courses increases or if there are requests for more functions in the future, it will be necessary to consider using the paid version, including budgeting.

References

[1] Mathematics, Data Science and AI Education Accreditation System, Ministry of Education, Culture, Sports, Science and Technology (MEXT) Official Web Site, https://www.mext.go.jp/a_menu/koutou/suuri_datascience_ai/00001.htm, Last Accessed on 2023.3.24.

[2] Support Aiming at "Establishment or Conversion of 250 Faculties" in Science, Engineering and Agriculture, MEXT's 10-year Plan, Yomiuri Shimbun Online, https://www.yomiuri.co.jp/kyoiku/kyoiku/news/20230112-OYT1T50041/, 2023.1.12.

[3] Miyamoto Y.: "An Approach of Programming Education Targeting Hybrid Mobile Application Development," in Proceedings of the 11th Annual - 2022 Science, Technology, Engineering, Arts, Math & Education Conference, Hawaii University International Conferences, Vol. HUIC-STEAM, No. 2022, pp. 1-6, 2022.

[4] Unity Real-Time Development Platform | 3D, 2D, VR & AR Engine, Unity Technologies, https://unity.com/.

[5] Suzuki M.: "Introduction to Unity: Learn by Making," Gijutsu-Hyoron Co., Ltd., 2019.

[6] Kojara: "[Unity 2021 Version] How to Apply Starter Assets," Nekojara City, https://nekojara.city/unity2021-starter-assets/, Last Updated on 2022.2.26.