

**ШУМЕНСКИ УНИВЕРСИТЕТ "ЕПИСКОП
КОНСТАНТИН ПРЕСЛАВСКИ"**

**SHUMEN UNIVERSITY "BISHOP KONSTANTIN
PRESLAVSKY"**

НАУЧНИ ТРУДОВЕ

КОЛЕЖ – ДОБРИЧ

ТОМ XIII

PROCEEDINGS

COLLEGE DOBRICH

VOLUME XIII



**Университетско издателство
"Епископ Константин Преславски"**

НАУЧНИ ТРУДОВЕ на Колеж - Добрич, том XIII, 2021 г.

РЕДАКЦИОННА КОЛЕГИЯ

проф. д.м.н. Николай Янков – отг.
редактор
проф. д-р Найдено Ненков
доц. д-р Милена Цанкова
доц. д-р Живка Илиева

Превод

доц. д-р Живка Илиева

Коректор

гл. ас. д-р Камелия Койчева

Оформление

проф. д.м.н. Николай Янков

Университетско издателство “Епископ Константин
Преславски”

Шумен, 2021 г.
ISSN 2367-8356

VR СИМУЛАТОР ЗА ОБУЧЕНИЕТО ПО ПРОГРАМИРАНЕ НА LEGO РОБОТИ

ПАВЕЛ Г. ГРАДИНАРОВ, НАЙДЕН В. НЕНКОВ

VR SIMULATOR FOR LEGO ROBOTS PROGRAMMING TRAINING

PAVEL G. GRADINAROV, NAYDEN V. NENKOV

ABSTRACT: *The article describes an application for developing a virtual simulator of Lego robots, which is used for training in their programming. The basic steps in using the C # language and the UNITY game engine in creating the simulator are shown. The application is designed to provide easy selection and management of the user interface of students and is adapted to work with virtual reality - VR, including a helmet. The idea is to simulate learners to use this type of technology and their affinity for learning robot programming. The functional possibilities of the created simulator during training are also described.*

KEYWORDS: *VR-virtual reality, simulator, 3d – objects, LEGO Mindstorms robot.*

Увод

Необходимостта от въвеждане в обучението на програмиране на роботи е безспорна. Усвояването на тези технологии е важно за всички категории обучаеми – ученици, студенти и любители, които желаят да се впишат в дигиталния свят на съвременността. Решаващо е за търсещите работа, които искат да имат съвременна и удовлетворяваща техните финансови и емоционални стремежи реализация [1, 3, 5].

В практиката се срещат различни видове роботи, които могат да се прилагат за целта на обучението. Предмет на статията е обучението по програмиране в университета или училището. Анализът на съществуващите решения показва, че LEGO платформата е подходяща за различни възрасти и възможности на

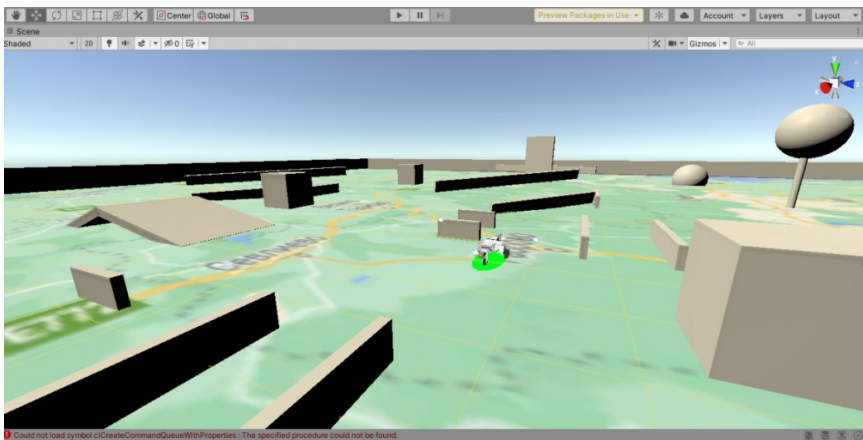
обучаващите се. При нея се отчитат доста бърз напредък и резултати за споменатите категории.

Недостатък е високата цена и невъзможността да се осигури на всеки обучаем комплект. Решението е разработването на симулатори, които позволяват на всеки, който притежава компютър или таблет, да ги използва [2]. Освен това симулаторът се комбинира с VR оборудване, което дава възможност за още по-атрактивно представяне на визуалното програмиране [12,13,14,15]. Целта е да се засили емоцията и желанието за усвояване на тези обещаващи технологии сред студентите и учениците. В изложението по-нататък се показват етапите на разработване на симулатора.

Изложение

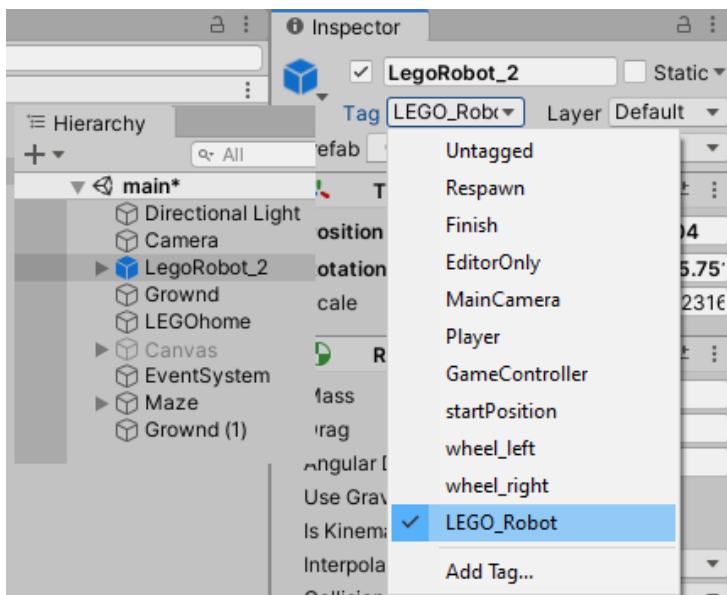
Интерфейсът на текущия VR симулатор е направен максимално близко до реалната среда за блоково програмиране на LEGO роботи, като са създадени различни препятствия, подходящи за обучението, наподобяващо отворен лабиринт. Роботът, с който ще бъде извършвано обучението [6, 7], също е построен максимално да наподобява реалния. Така се допълва усещането за реалност (виж фиг. 1).

Идеята е да се създаде непретенциозна среда за изучаване командите за управление на робота, които се програмират с визуални блокове много близки до реалните. За тази цел се използва MS Visual Studio, програмният език C# и междуплатформената среда UNIT.



Фиг. 1. Виртуален свят на симулатора на LegoRobot EV3.

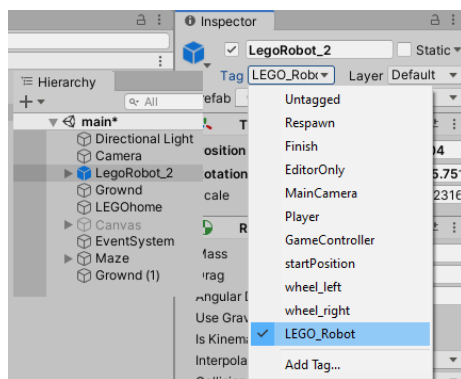
3D обектите се намират в Йерархията, като всички са обединени в един родителски обект „LegoRobot“ (виж фиг. 2).



Фиг.2 Класът LegoRobot

Родителският клас **LegoRobot** съдържа множество от обекти, които се явяват като родители на други обекти от проекта. Имената съответстват на предназначението на обектите. Така например при евентуално желание да се промени роботът като функционалност или как да изглежда, няма да е необходимо да се изгражда целият робот.

Трябва да се изтрие даден компонент или да се премести в Scene. За да е надеждна програмата за управление на обектите, е създадена гъвкава организация на проекта. Тя се изразява във факта, че всеки C# скрипт, отговарящ за дадена функция от робота е дефиниран в съответния родителски обект. В проекта също е използван и друг подход чрез използване на тагове към определените обекти с команда: „`GameObject thisObject = GameObject.FindGameObjectWithTag(“LEGO_Robot”);`“ (виж. фиг. 3).



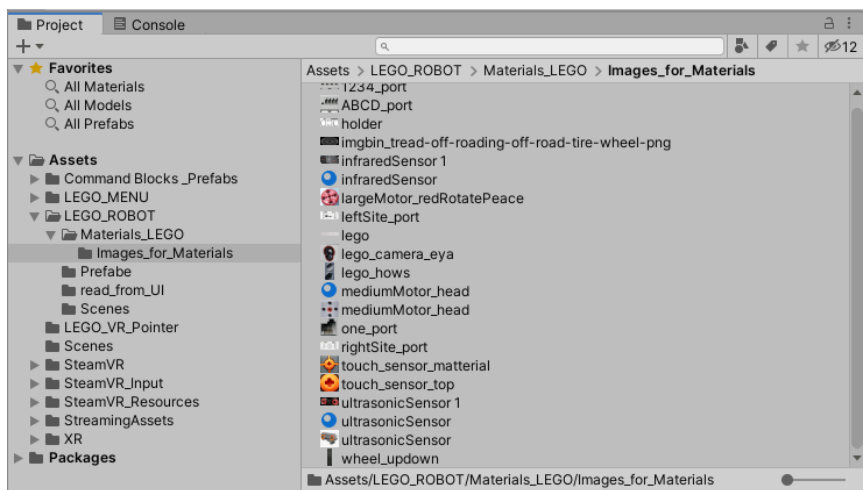
Фиг. 3. Промяна на Tag на LegoRobot_2

Първата стъпка е създаването на папки за съхраняване на компонентите. За създаването на Lego робота е създадена папка **LEGO_ROBOT**, където се съхраняват всички подпапки с различно съдържание. Създадени са още няколко папки -> **Materials LEGO** , **Prefabs** и още няколко, които служат за работни и не са

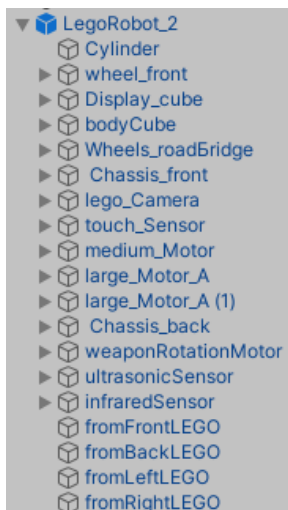
нужни за този проект. В папката Materials се намира папка Images, където се съхраняват изображенията, използвани от Material компонента.(виж фиг. 4)

По този начин е изграден всеки един материал, който се използва в проекта. Организацията на файловете е изключително важна заради евентуалната промяна, която може да доведе до загуба на компоненти и грешки в програмата. 3D обектите се намират в Йерархията, като всички са обединени в един родителски обект LegoRobot.(Фиг. 5)

Изграждането на робота се извършва като се използва снимков материал на реалния робот. На програма за графично моделиране се очертават правилните контури на изображенията и след това се добавят към проекта в папка Images, която се намира в папка LEGO_ROBOT. Изображенията се поставят в компоненти от тип material и след това този компонент се включва определения 3D обект.



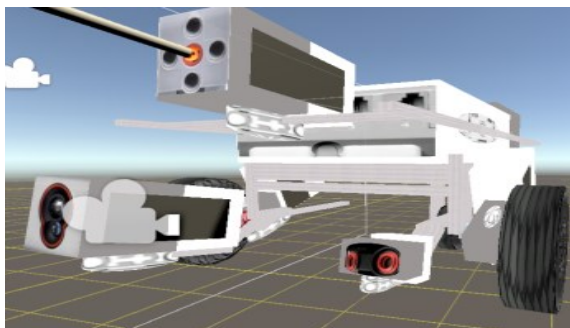
Фиг. 4 Папки и файлове на LegoRobot



Фиг.5 Обединяване на компонентите на робота в LegoRobot

От Asset Store на Unity3D са използвани материали за визуализиране на колелата. Добавен е дисплей както и бутони на командния модул. В настоящата разработка няма да бъдат засегнати до голяма степен възможностите нито на LEGO робота, нито на блоковите команди, а ще бъде разяснено главните и най-често използваните функции в „VR симулатор за LEGO робот“.

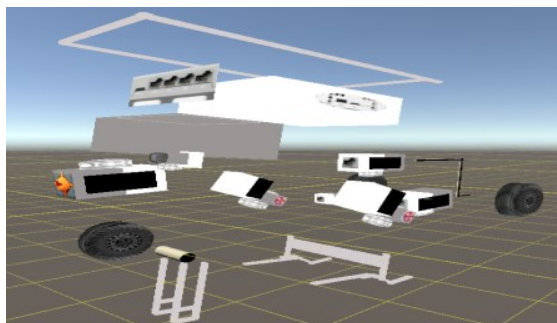
На фиг. 6 и 7 е изобразен LEGO роботът отпред и отзад, а на фиг. 8 роботът е разделен на главни компоненти.



Фиг. 6 LEGO от предната страна



Фиг.7 LEGO от задната страна



Фиг. 8 LEGO, разглобен на главни съставни компоненти

На фиг. 9 са изобразени класовете на робота. Вижда се как всяка част е цяла и напълно завършена и може да

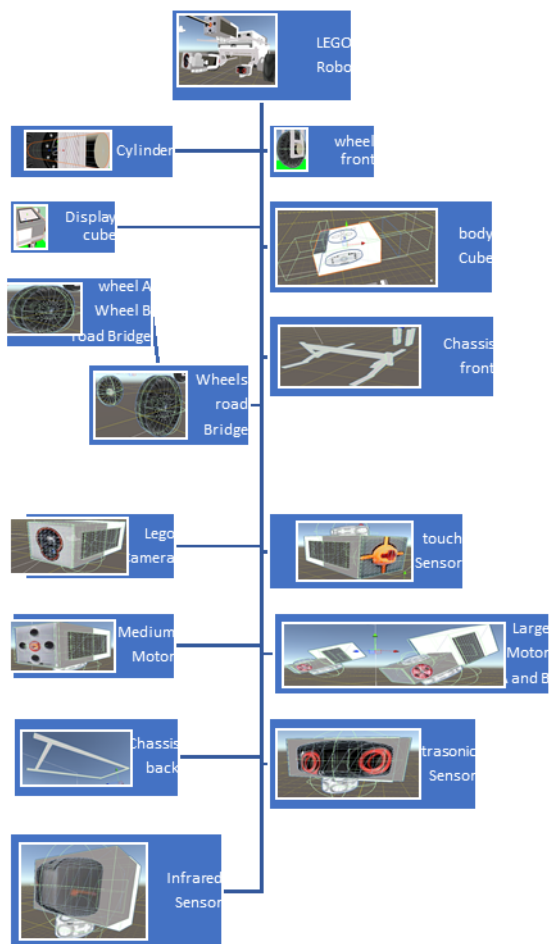
работи отделно от своя обект-родител. Идеята тук е да може да се остави възможност за по-нататъшна работа и изграждане на други прототипи на робота.

Придвижването на виртуалния робот се извършва по изчислени координати, а не от колелата. Причината е в това, че колелата се движат на място, те не извършват физическа сила върху обекта (Виж фиг. 10).

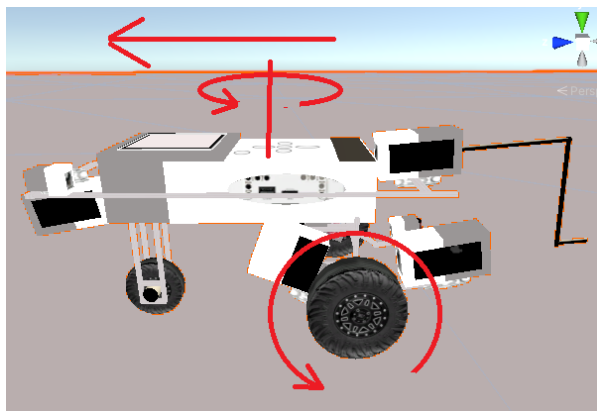
Колелата се въртят, но C# скриптът управлява робота дали да се завърта, дали да обикаля около даден обект, дали да се движи по права и след колко време да спре.

Двата вида мотори Medium Motor и Large в реалната среда се различават по това, че единият има по-бързо завъртане, но е по-слаб (Medium Motor) а другият има по-бавно завъртане (Large Motor), но е по-мощен. Има различни възможности да се построи LEGO роботът, които варианти позволяват и различни действия на робота като захващане стрелба и т.н. В реалната среда Medium Motor се използва именно за оръжие или нещо помощно. Докато Large Motor се използва за движение на целия робот.

В този проект разлики в движенията или мощността на моторите не са поставени. Използват се два броя Large Motor за задвижващите колела и един Medium Motor е поставен (статично) за използване на оръжие – подобие на манивела, която може да избутва препятствия. В програмата се създават Prefabs (заготовки), които служат за многократна употреба, също като класовете в обектноориентираните езици.



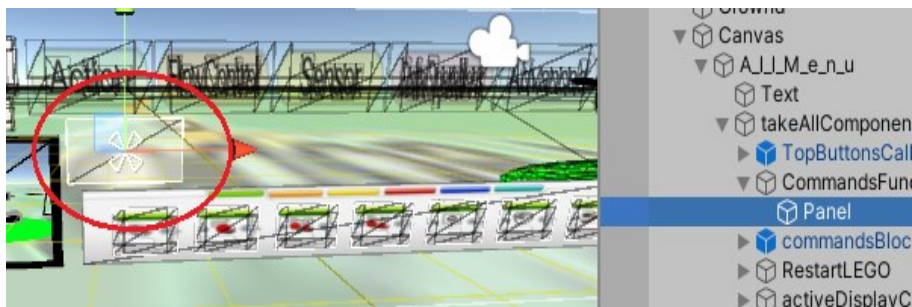
Фиг. 9 Класове на робота LEGO в симулатора



Фиг. 10 Движение на LEGO робота

Изграждане на LEGO Менюто

Менюто, което се използва за задаване на команди на виртуалния робот е почти идентично като това, което се използва при програмиране на LEGO. Използвани са същите цветове и форми, както и начини на задаване на командите. Не е пресъздадено нареждането на блоковете чрез приплъзване като в реалната среда, за да не се натовари програмата, затова те се извикват чрез натискане на бутони. Има предварително зададена стартова позиция (виж фиг 11) и при всяко следващо извикване на команден блок, позицията се премества надясно. Това не дава свобода на потребителя на симулатора да размества блоковете, което по принцип е възможно и е много удобно. За да се коригира потокът от блокове, те може да се изтрият последователно отзад.







Фиг. 11 Стартова позиция за командните блокове

Причината е и начинът, по който се изпълняват, тъй като при добавяне на даден блок се добавя в контейнер. Той определя и последователността на изпълнение след това на командите.

В реализацията тук са налични само 4 командни блока за движение и един за пауза. Четирите командни блока са от Action Block менюто: MediumMotor, LargeMotor, MoveSteering и MoveTank. Блокът за пауза Wait е от Flow Control (Виж таб. 1).

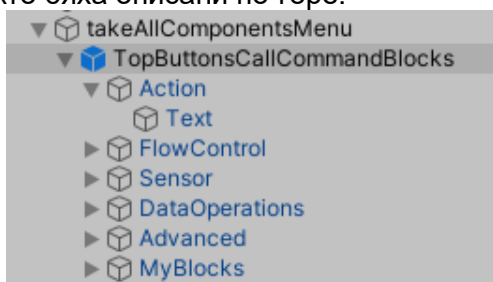
Таблица 1 Изобразяване на 4-те екшън команди

Medium Motor	Large Motor	Move Steering	Move Tank
- Off	- Off	- Off	- Off
- On	- On	- On	- On
- OnForSeconds	- OnForSeconds	- OnForSeconds	- OnForSeconds
- OnForDegrees	- OnForDegrees	- OnForDegrees	- OnForDegrees
- OnForRotations	- OnForRotations	- OnForRotations	- OnForRotations
			

С тези команди ще се даде началото на тестването на работата. Освен да се свикне и да се възприеме VR средата, тези команди са и достатъчни, за да се усвоят

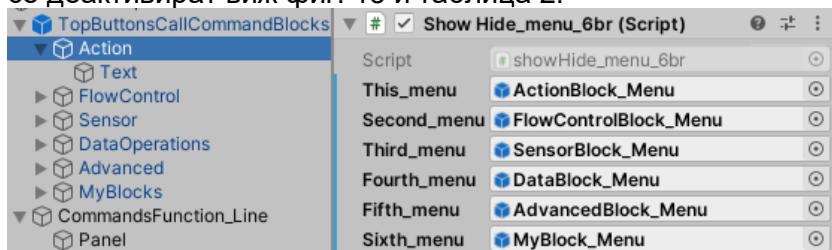
фундаменталните страни на движение, които са абсолютно същите като в реалния свят.

Unity3D позволява Потребителският интерфейс UI да бъде изобразен по няколко начина. В тази разработка се използват създадени пет обекта, които да се грижат за съхранение на информацията на менюто. Четирите обекта се съхраняват в един главен обект, който може да се активира и деактивира чрез натискане на един стационарен червен бутон намиращ се горе вляво с надпис „MENU“. TopButtonsCallCommandBlocks (Виж фиг. 4.17) Тук се съхраняват шестте бутона, които извикват командните блокове, както бяха описани по-горе.



Фиг. 12 Обект, съхраняващ бутоните за извикване на командните блокове

Към всеки един от тези бутони е включен C# скрипт. Той позволява да се активира изборият, а останалите да се деактивират виж фиг. 13 и таблица 2.

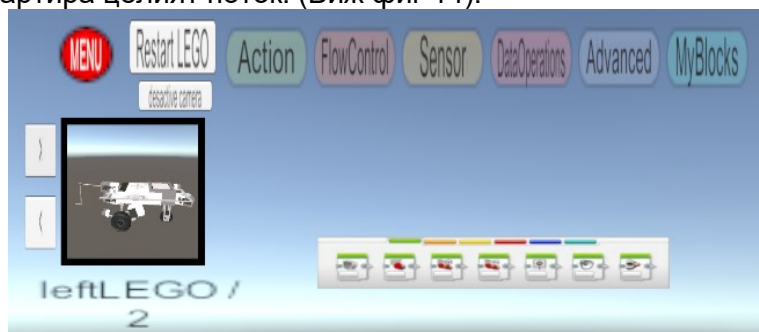


Фиг. 13 Прикачен C# script към обект

Таблица 2. Скрипт за управление

<pre> using System.Collections; using System.Collections.Generic; using UnityEngine; using UnityEngine.EventSystems; public class showHide_menu_6br : MonoBehaviour, IPointerClickHandler { public GameObject this_menu; public GameObject second_menu; public GameObject third_menu; public GameObject fourth_menu; public GameObject fifth_menu; public GameObject sixth_menu; // Start is called before the first frame update void Start() { his_menu.SetActive(false); second_menu.SetActive(false); third_menu.SetActive(false); fourth_menu.SetActive(false); fifth_menu.SetActive(false); sixth_menu.SetActive(false); } // Update is called once per frame </pre>	<pre> public void OnClick(PointerEventData eventData) { if (!this_menu.activeSelf) { this_menu.SetActive(true); if (second_menu.activeSelf) second_menu.SetActive(false); if (third_menu.activeSelf) third_menu.SetActive(false); if (fourth_menu) fourth_menu.SetActive(false); if (fifth_menu) fifth_menu.SetActive(false); if (sixth_menu) sixth_menu.SetActive(false); } else { this_menu.SetActive(false); } }} </pre>
--	---

Освен появяването на шестте бутона горе за управления на различните действия на робота, менюто ни дава възможност да включим външна камера, която да следи робота от разстояние. Камерите, които можем да избираме са четири на брой. Така, ако изпратим робота надалеч, можем да следим околната среда. Това е добър ефект към интерфейса, а и придава по-завършен вид на менюто. За улеснение е добавено и текстово поле под въпросния екран, за да се вижда коя камера е активна. Чрез START бутона, който се появява автоматично се стартира целият поток. (Виж фиг 14).



Фиг. 14. Интерфейс на VR менюто

Начинът на работа на Старт бутона при вече наредени блокове е следният: използват се 2 броя контейнери от тип `Queue<GameObject>`. Когато добавяме обект към потока, той автоматично се добавя към 1-вата опашка. Така тази опашка се пълни постоянно.

При натискане на бутон за стартиране на програмата, блокът който е първи, се записва (копира) във втория контейнер и се изтрива от първия. Във втория контейнер започва изпълнението му. Когато приключи булева променлива „endTask“ си променя стойността на „true“ и обектът се изтрива. Тогава се дава команда да се копира следващият обект (команда от първия контейнер). След като се копира на втория, се изтрива от първия. Проверява се дали вторият контейнер не е празен. Ако не е празен, тогава се дава команда за стартиране на задачата.

Действията продължават докато първият и вторият контейнер не се освободят от информация.

Важно. Когато се извикват командните блокове, те се нареждат в потока. Има помощен масив, който при стартиране на изпълнението на командите да копира целия първи контейнер. Проблемът е, че при изпълнение на задачата потребителският интерфейс остава пълен с наредени команди, но програмата ще бъде с празна памет, заради празния контейнер. Затова помощният масив ще се използва за връщане на цялата наредена информация в първия контейнер след приключване на задачата. Това ще стане, когато след проверката се установи, че и двата контейнера са се освободили от обектите в тях.

Работа с VR симулатора

След като потребителят включи симулатора, се налага да се включи и SteamVR, което означава че компютърът трябва да има интернет. Пред потребителя се появява почти същият изглед, какъвто се вижда на десктопа с тази разлика, че е по-разтеглено настрани и във височина и, разбира се, в дълбочина.

Показалецът на контролера, който настройвахме, работи през цялото време. И се вижда как краят на показалеца се спира върху всеки обект.

Горе вдясно се забелязва един червен обект, на който пише MENU. Насочваме показалеца върху обекта и с бутона, намиращ се отгоре на контролера във формата на диск, натискаме обект. Това всъщност е главният бутон за показване и скриване на менюто.

Веднага след това се показва цялото контролно меню, шестте варианта за извикване на командните блокове както и още два бутона в началото. Единият рестартира местоположението на робота. Другият включва изгледа на четирите камери, които са прикрепени към LEGO робота. С две стрелки можем да сменяме изгледа на камерите. Отляво, отдясно, отпред и отзад.

При избиране на Action се появява лента с менюта отдолу, която е същата като при истинското блоково програмиране на LEGO робота.

Избираме MediumMotor и се появява команден блок. Промяната на параметрите е по същия начин както и в оригинал, само че в случая се избира с VR показалец.

Когато изберем въпросния блок или комбинация от блокове натискаме бутона START, при което програмата започва своята работа като задейства робота.

Можем по всяко време да върнем робота в стартова позиция като натиснем бутона RESTARTLEGO.

В симулатора има поставени препятствия с цел да се мотивира потребителят да прави различни комбинации, за да може да ги заобиколи и премине. Тук се използва въображение и преценка на разстоянието.

Заклучение

В симулатора има много действия, които трябва да се добавят като движение на потребителя, както и да се завършат всички команди. Да се добави симулация на околната среда като ден и нощ, с цел да се програмира робота да извършва определени действия при затъмняване или при активиране на светлината. Да се добави функция за събиране на данни за околната среда (къде има обекти и да ги картографира).

Експериментът с виртуалната среда и със симулатора показва добро възприемане от обучаемите студенти и ученици и положителни емоции, които се констатира при използването му. Това са и насоките за по-нататъшната работа по този проект.

ЛИТЕРАТУРА:

1. **С. Нанков , В.Колев**, колектив, Въведение в програмирането със C# (2018)ISBN 978-619-00-0778-4
2. **Д. Минчев**, Жълта книга по C# (2013) ISBN 978-954-9925-84-5.

3. **Д. Минчев, А. Иванов, РЪКОВОДСТВО ПО РОБОТИКА. LEGO MINDSTORM NXT 2 КОМПЛЕКТИ 9797 И 9695, 2015,** Бургаски свободен университет ISBN 978-619-7126-18-1
4. **M. Menard, B. Wagstaff, Game Development with Unity-second Edition (2014)ISBN – 1-305-11054-4**
5. **V. Gerasimov, D.Kraczla, Unity 3.x Scripting (2012) ISBN – 978-1-84969-230-4**
6. **T. Norton, Learning C# by Developing Games with Unity 3D (2013) ISBN-978-1-84969-658-6**
7. **Thorn, Mastering Unity Scripting (2015) ISBN-978-1-784-39-065-5**
8. **M. Geig, Unity 2018 Game Development in 24 Hours (2018) ISBN-978-13-499813-8**
9. **Y. Isogawa, The LEGO MINDSTORMS EV3 idea book ISBN-978-1-59927-600-3**
10. **Dean, Unity Character Animation with Mecanim (2015) ISBN-978-1-84969-4**
11. **S. Jackson, Unity 3D UI Essentials (2015)ISBN-978-1-78355-361-7**
12. **M. Smith, Unity 4.x Cookbook (2013) ISBN-978-1-84969-042-3**
13. Unity: <https://unity.com/pages/unity-pro-buy-now> [посетен на 12.06.2021]
14. Unity - <https://unity.com> [посетен на 10.07.2021]
15. Unity - <https://www.youtube.com/user/Unity3D> [посетен на 10.07.2021]

Павел Георгиев Градинаров е дипломант в специалността „Софтуерни технологии“ на магистърската програма „Информатика“ към ФМИ на ШУ “Епископ К. Преславски“: pavel.gradinarov@gmail.com
 проф. д-р Найден Вълков Ненков е преподавател в Колеж Добрич: n.nenkov@shu.bg