Lucrare de atestat

Paradox

Aplicație

**Elev:** Takacs Darius

**Clasa:** a XII-a B

**Profil:** Matematică-Informatică, intensiv Informatică

**Profesor coordonator:** prof. Lőkös Annamária

**Motivatie si Initiere**

Motivul principal pentru aceasta lucrare este simplu. Experimentare, autoimbunatatire si crearea acestui atestat.

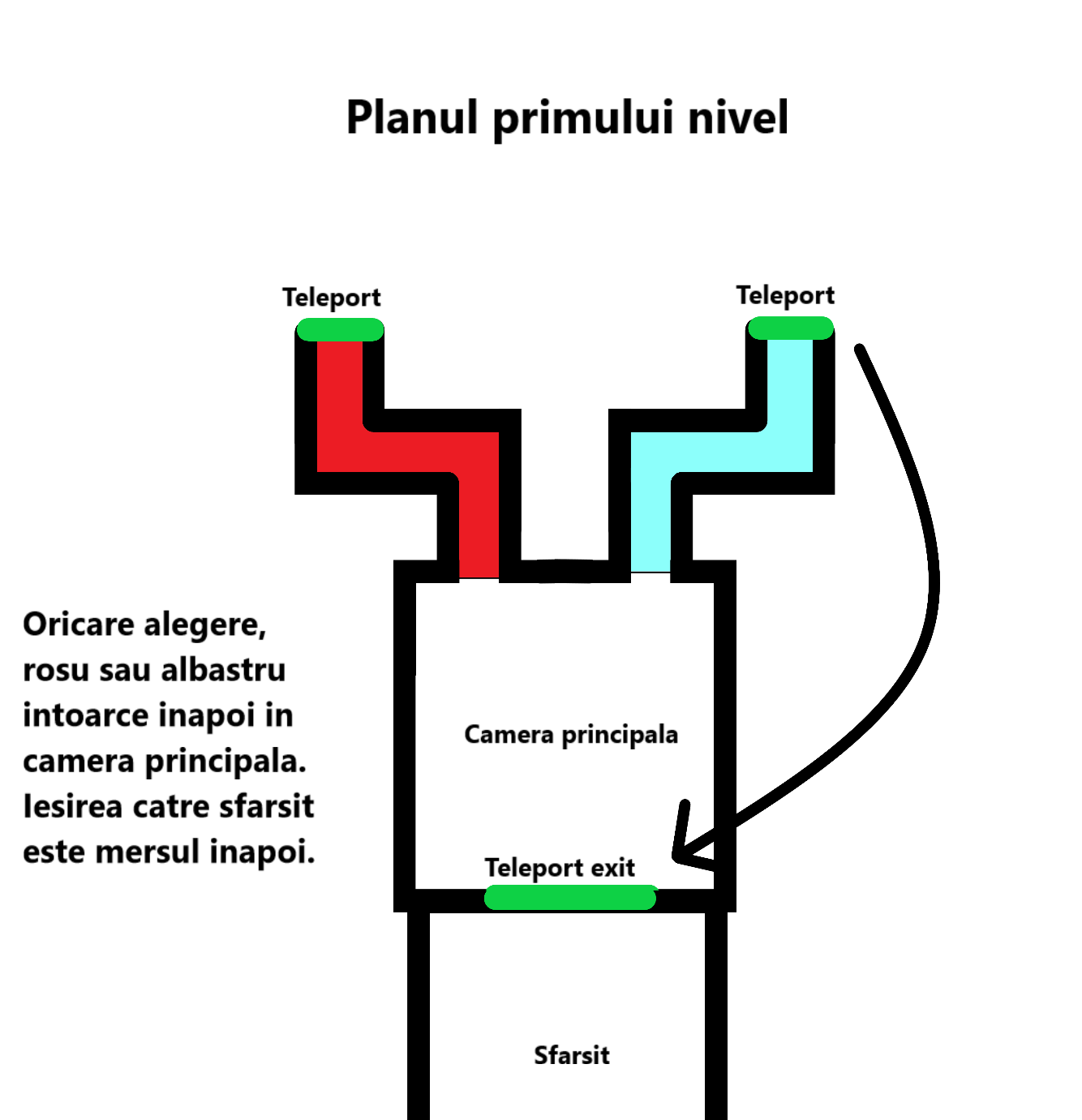
Ideea initiala a proiectului a inceput din inspiratia a mai multor jocuri. In principal Antichamber de pe Steam si This Level Again de pe android. Pentru proiect a fost alesa paltforma android deoarece din prima am vrut sa folosesc mecanici foarte subestimate ale telefoanelor. Precum giroscopul pentru detectarea vitezei telefonului, camera pentru a filma lucruri diferite (in cazul de fata foarte simplu culoarea albastra).

Proiectul nu se poate denumi un joc deoarece este mai mult un tech demo. O prezentare mai interesanta de mecanici care pot fi folosite pentru crearea unor jocuri complete. Aceasta tematica de puzzle cu mecanici diferite pe nivel este mult prea complex pentru a functiona in sine. **Paradox** (proiectul) ofera o senzatie de „ce programare interesanta si originala, is curios ce altceva mai are”,nu doar ca aceasta metoda trebuie sa fie foarte mult prelungit, cea ce e foarte greu, ea nici nu ofera senzatia dorita de „eu am reusit sa trec acest nivel folosind mecanicile in mod inteligent”. A 2-a situatie face jucatorul sa se simta mandru, prima face sa se simta impresionat. In general a 2-a situatie este mai buna.

**Inceputul**

Proiectul a fost facut in totalitate in platforma Unity in limbajul c#.

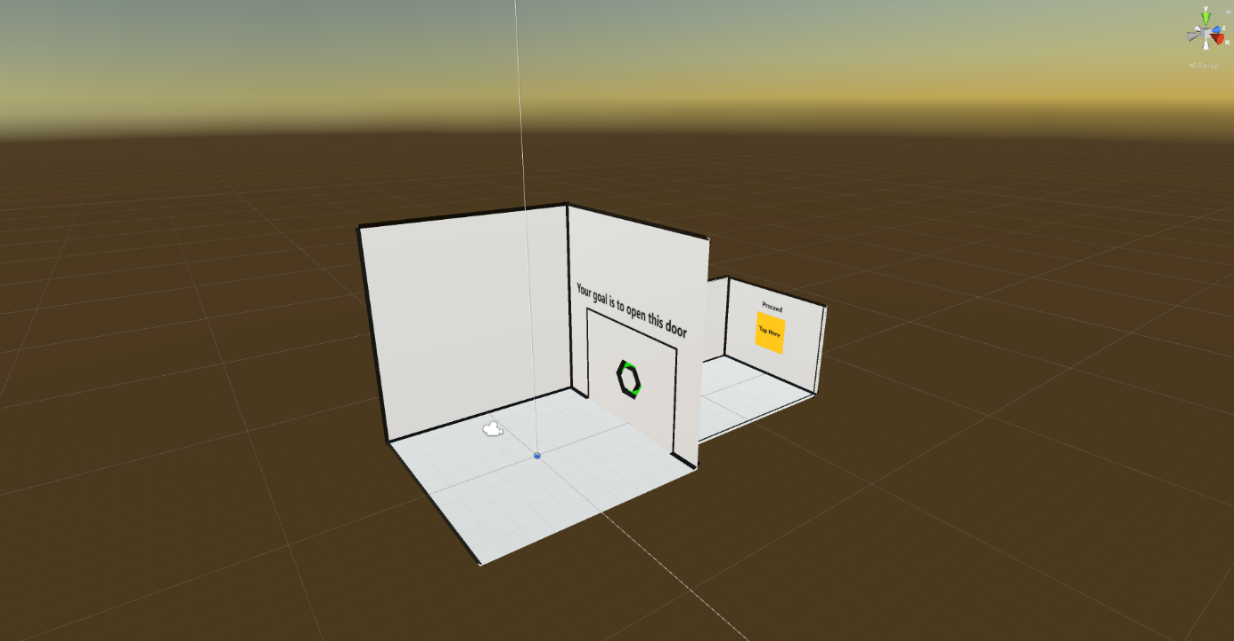
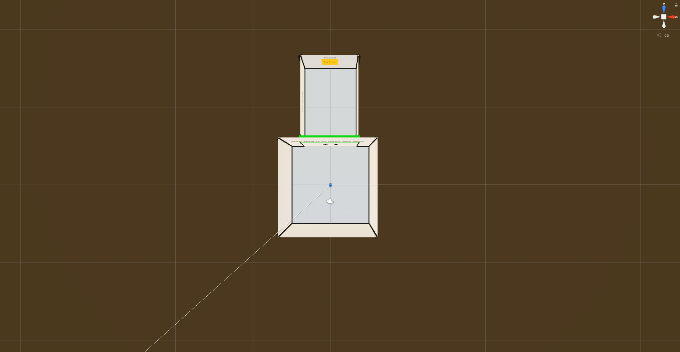
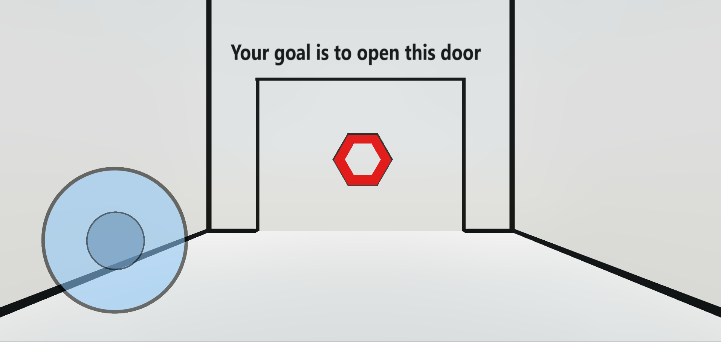
La inceput **Paradox** a aratat total diferit. A avut o inspiratie mult mai mare din Antichamber, fiind tot 3d am inceput imitand niste nivele din Antichamber. Am observat blocajul rapid in aceasta metoda si am abandonat-o dupa primul nivel



Aceasta idee este foarte asemanatoare cu cea din Antichamber. A fost abandonata foarte repede deoarece nu se putea continua si nu avea acea repetite a aceeasi incaperi. Lucru care a fost dorit din prima.

Repetarea aceleasi incaperi are scopul de a pune accentul pe detaliile mici care se schimba si mecanicile. Aratand ca pentru aceasi problema in aparenta exista foarte multe solutii in esenta. Repetarea mai are si un efect de prelungire, ea evidentiind mecanicile jocului mai mult, in aceasi incapere intamplandu-se toate diferitele mecanici intrigante.

De acea dupa acel esec sa ajuns la o camera mare patrata cu o usa in fata jucatorului si o singura iesire. Dupa care se revine la camera.

.

Camera patrata a fost alesa pentru a avea posibilitatea de a ascunde solutii pe acoperis, lucru la care nu oricine se uita initial. Fiecare camera are instructiuni deasupra usii pentru a indruma jucatorul catre solutia acelui nivel.

Camera este foarte mare iar jucatorul are tot timpul sa o exploreze, dar nu are nici un motiv sa o faca. Acest lucru a fost schimbat in primul nivel (dupa intro cel din poza).

Intro-ul in joc nu are nici o provocare este pur si simplu pus acolo pentru a arata incaperea si miscarea usii, si pentru a face jucatorul sa se adapteze controalelor. Nu este pus un meniu de instructiuni la start, nici un meniu de setari si start. Acest lucru e intentionat deoarece din prima am dorit ca sa evidentiez experienta ca fiind diferita de cea ce se asteapta normal un jucator de la ceva asemanator. Joystickul din stanga controleaza miscarea, iar intoarcerea camerei se face pe toata partea dreapta swipe.

**Paradox** mai are si 2 butoane specifice cu care se interactioneaza diferit „Collide Here” si „Tap Here”



Butonul „Tap Here” interactioneaza cu touch, trebuie sa apese cu degetul pentru a-l activa. Butonul „Collide Here” functioneaza cu personajul din lumea virtuala. Trebuie sa intre in el pentru a-l activa. Aceasta diferenta este folosita in mai multe aspecte mai tarziu in proiect.

**Paradox** mai are si o mecanica de autoajustare comform nivelului de capacitate a jucatorului sau. Adica daca observa ca un jucator este blocat instructiunile deasupra usii se vor schimba pentru a-l ajuta mai mult. Sistemul e foarte limitat, el doar masurand timpul din nivelul actual fara sa ia in considerare nivelul precedent. Un sistem mai adecvat ar fi unul de inteligenta artificiala care a fost invatata sa se adaptaze la miscarea si timpul jucatorului pentru a arata instructiunea potrivita direct din start.

**Nivelul 1**

Primul nivel este o idee inspirata din jocul Antichamber, o camera si o usa rusinoasa. Umorul din joc este libertatea mea artistica, dar consider ca i se potriveste tematicii neconventionale.

Cum am precizat acum cateva paragrafe primul nivel are un scop de familiarizare a camerei, si a tematici.

Aici jucatorul trebuie sa observe faptul ca uitandu-se la usa o rusineaza, se inroseste si se inchide, dar uitandu-se in alta directie usa se deschide. Provocarea vine in a-si da seama ca poate merge cu spatele prin usa pentru a progresa.

Acest nivel prezinta si un easter-egg, care este pus pentru a evita o problema si pentru a incuraja exploratul si mintile cu adevarat briliante. Accesul la acest easter- egg se face prin coborarea usii, dupa care jucatorul se pune deasupra usii coborate si se uita la ea, aceasta se ridica si il va duce prin pereti intr-o camera neagra cu un mic text de felicitare. Asa s-a si umplut si negative-posibility-space, si s-a evitat problema bug-ului incorporandul in joc.



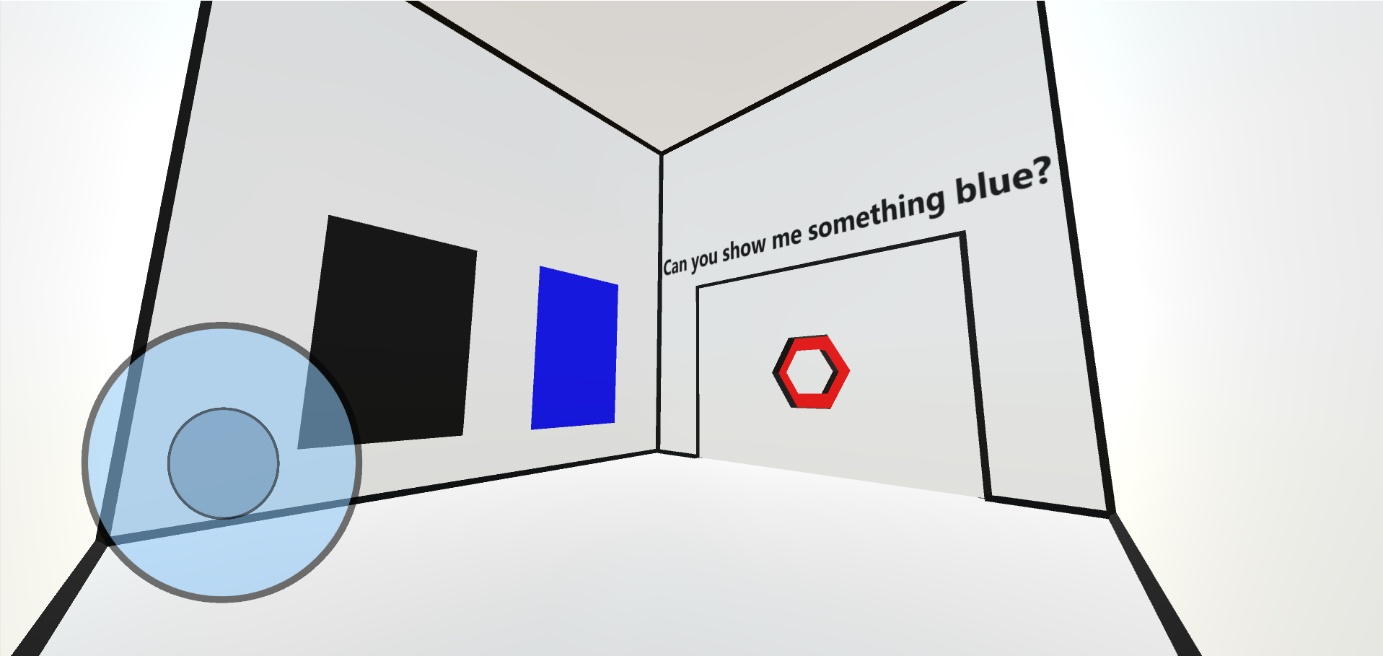
Aceasta metoda de a crea easter-egg din buguri nu se poate cu orice bug. Trebuie sa implineasca cateva criterii.

* Este foarte restrans, se aplica doar pe o specifica parte a jocului.
* Este foarte previzibil, se comporta la fel de fiecare data cand se executa aceasi actiune.
* Nu se contrazice cu mecanicile principale ale jocului.

Bugul acesta s-a potrivit perfect tuturor criteriilor.

**Nivelul 2**

Acest nivel prezinta prima schimbare a camerei si prima patrundere in lumea reala. Toate nivelele au instructiunile deasupra usii.



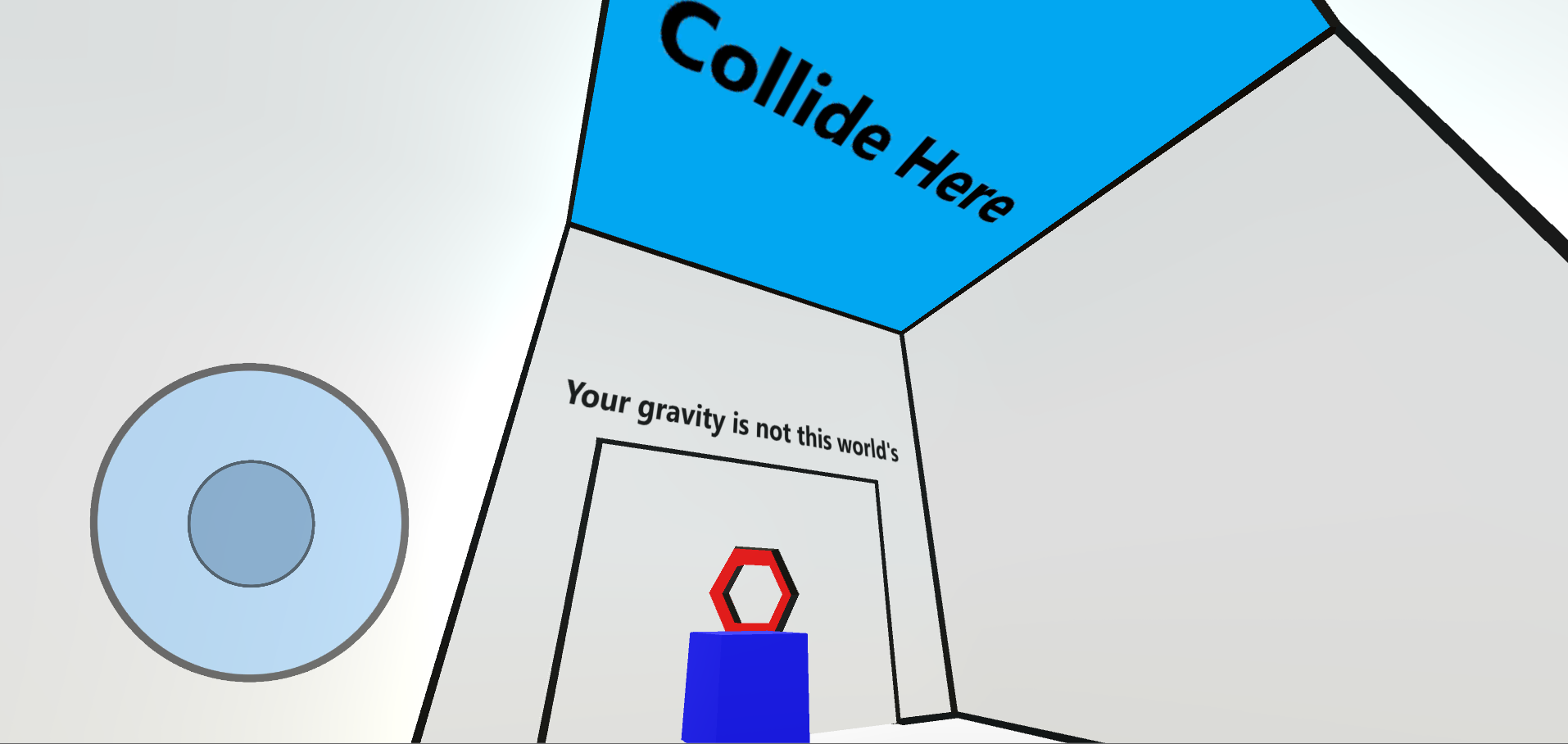
Scopul acestui nivel este cum am spus anterior patrunderea in lumea reala. Primul panou (negru) este de fapt imaginea camerei telefonului. Prima mecanica intriganta a telefonului. (Motivul pentru care e negru in poza este pentru ca poza a fost facuta pe calculator, pentru calitate marita)

Provocarea prezentata in acest nivel este gasirea unui obiect care sa se asemene cu panoul din dreapta (albastru). Acest lucru va deschide usa.

Din play-testul micut al lui **Paradox** am conclus ca acesta era cel mai popular nivel. Asa ca l-am adus mai aproape de inceput nivel 2 nu 5 cum a fost initial.

**Nivelul 3**

Nivelul 3 e singurul care se foloseste de acoperisul foarte inalt al camerei pentru a ascunde solutia. Si acesta patrunde in lumea reala, dar comparativ cu cel anterior el patrunde mai mult fizic.

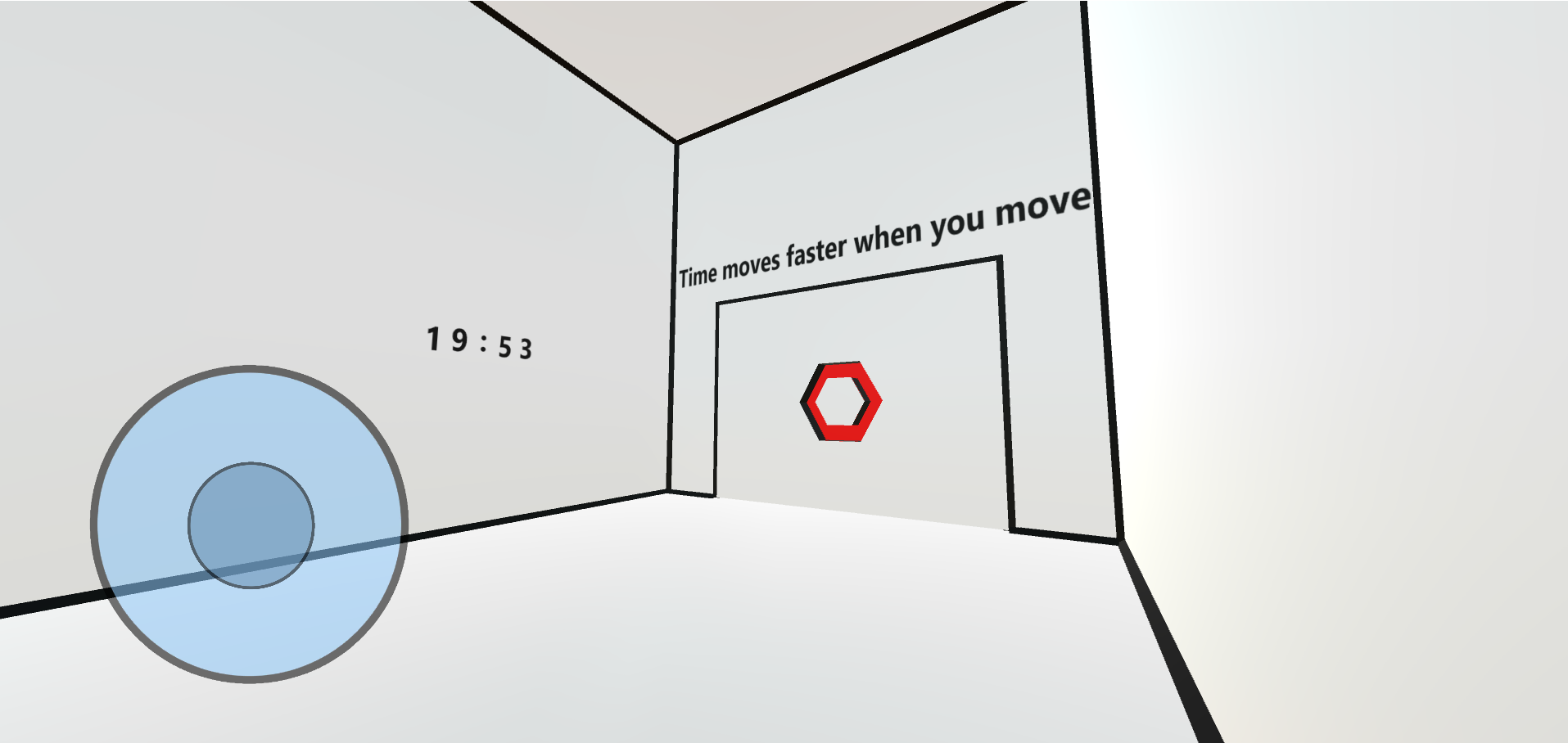


Scopul acestui nivel este de a explora mecanicile telefonului si de a intriga jucatorul cu idea de 2 lumi, cea virtuala a lui **Paradox** si cea reala, in care se afla el. Cubul albastru este afectat de gravitatie, dar nu de cea a proiectului ci de cea a realitati.

Provocarea este in realizarea mecanici neintuitive, si folosirea acelei mecanici pentru a deschide usa. Butonul „Collide Here” interactioneaza si cu cubul, acest lucru e destul de evident.

Acesta e primul nivel in care dificultatea lui **Paradox** incepe sa fie evidentiata.

**Nivelul 4**

Acest nivel se foloseste de fizica ca si cel anterior, dar de data asta el se foloseste de fizica mai complexa (relativitatea lui Einstein). Prezinta cea mai mica modificare, un ceas.

Scopul acestui nivel este progres de pe cel anterior. Daca in el s-a cerut fizica elementara de miscare si gravitatie, acesta cere fizica mai avansata in miscarea spatiu-timp.

Cea mai mare dificultate a acestui nivel este lipsa de cunoastere a marii majoritati a jucatorilor, nestiind ca marind viteza, timpul trece mai repede. Evident acest efect a fost accelerat enorm in **Paradox**, deoarece efectul in realitate necesita viteze apropiate de cea a luminii.

Nivelul 4 are si o solutie neeleganta, asteptarea intregului timp de 20 min. Aceasta „reusita” nu rezulta in easter-egg, deoarece nu consider promovarea lenii o manevra buna.

**Nivelul 5**

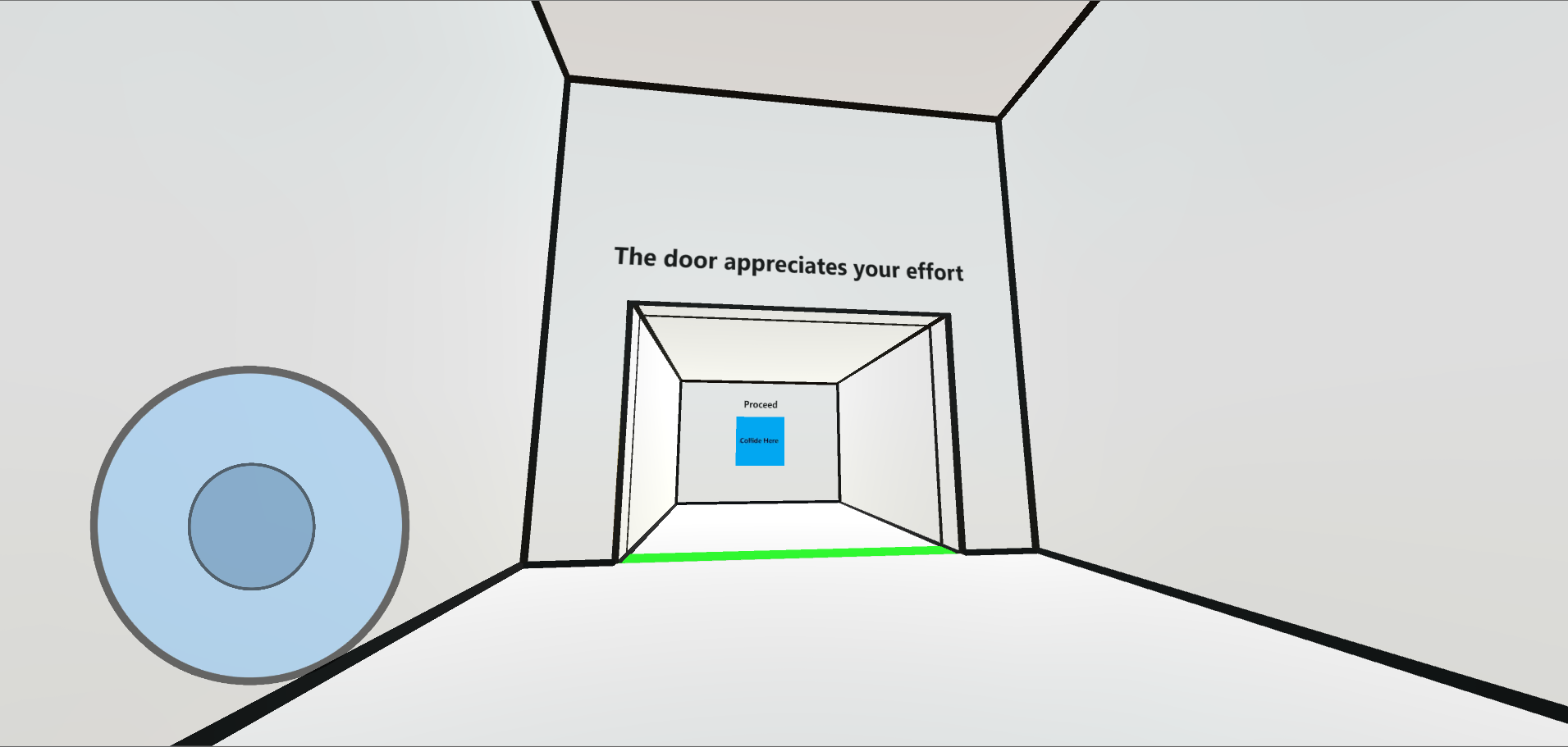
Nivelul 5 prezinta un pas inapoi in detaliile din camera, lucru intentionat pentru a mentine intriga. Acesta e si cel mai greu nivel dupa cum spuneau toti care l-au incercat.

Motivul existentei acestui nivel este de a reintroduce mecanicile virtuale. Acesta nu necesita patrunderea in realitate pentru solutie. In schimb necesita o gandire foarte creativa si o abordare diferita a problemei.

Solutia acestui nivel este foarte simpla dar foarte neintuitiva. Pentru a deschide usa jucatorul trebuie sa iasa si sa reintre in aplicatie.

Acest nivel a nimicit mintile jucatorilor necesitand o solutie care la foarte multi li se pare contraproductiva, de ce ai vrea sa il obligi pe jucator sa iasa afara. Eu sunt de parere ca exact acea intriga prin care **Paradox** sparge toate barierele predefinite despre jocuri este motivul existentei unui asemenea nivel.

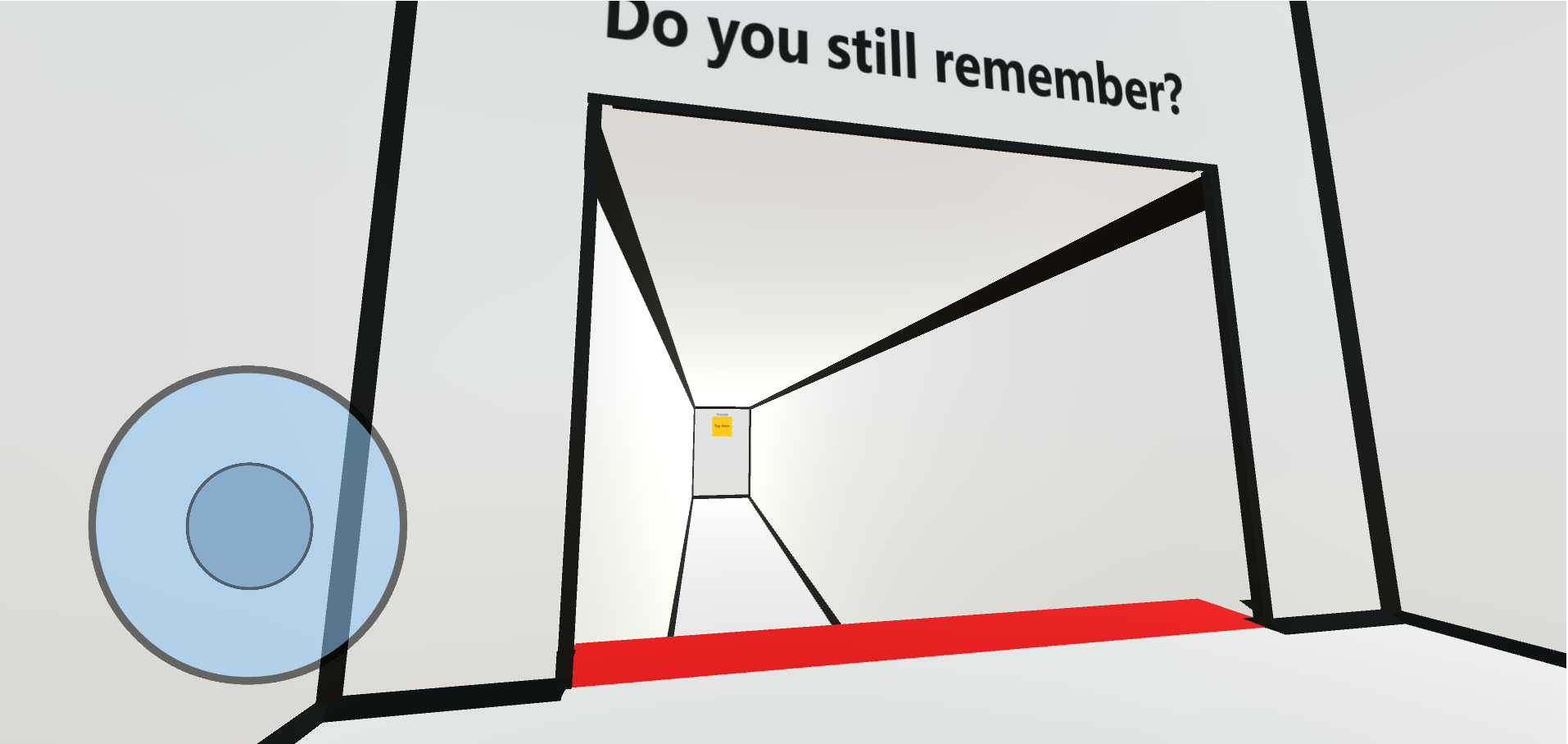
Nivelul 5 a fost initial nivelul 2 dar din cauza dificultati si popularitati nivelului 5 actual (cel cu camera telefonului) am facut schimbarea intre ele. Acest lucru nu a facut acest nivel cu nimic mai usor, chiar daca jucatorul a format mai multa experienta cu mecanicile neintuitive.



Usa deschisa dupa reintrarea in joc, animatia inca exista dar nu poate fi reprezentata in poza.

**Nivelul 6**

Penultimul nivel are cea mai mare modificare a camerei din tot proiectul. Doar aici usa incepe direct deschisa.

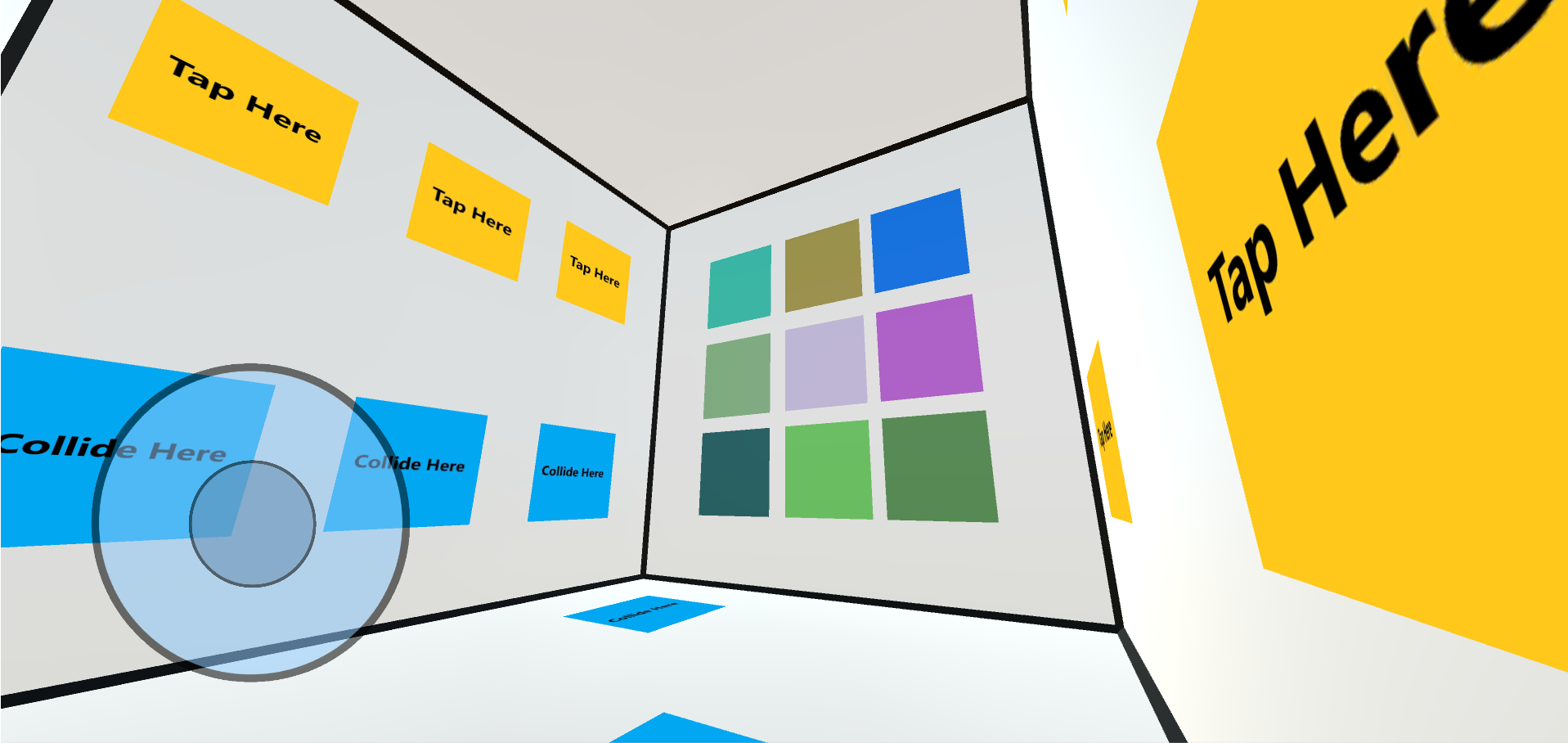
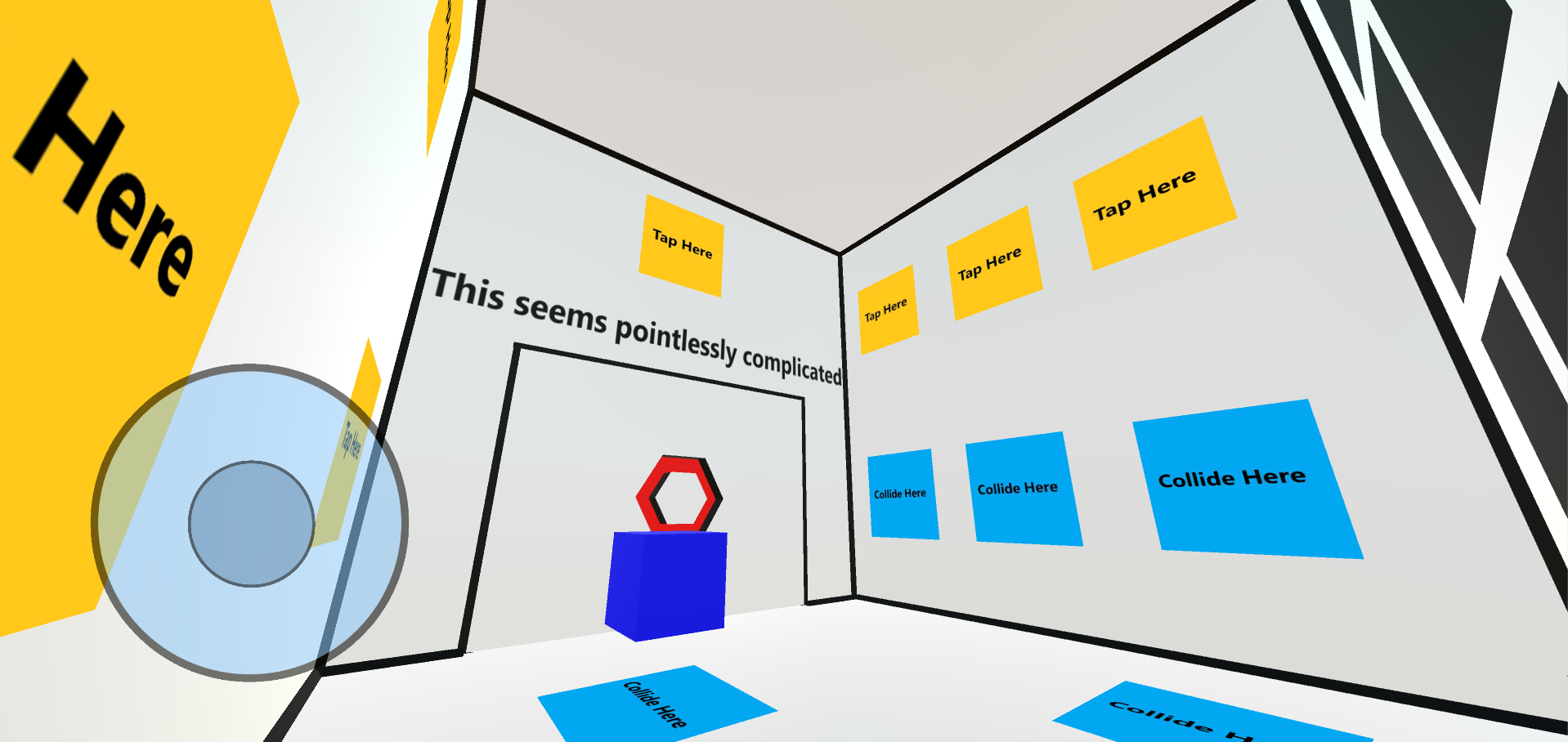


Scopul acestui nivel este de reutilizare a unor mecanici care cand au fost introduse pareau triviale. Atentia la detalii. Butonul din spate este galben „Tap Here”. Podeaua de dupa usa teleporteaza jucatorul inapoi la pozitia initiala.

Solutia nivelului este simpla, trebuie sa observe diferenta dintre acest buton si toate butoanele „Collide Here” anterioare. Dupa care doar apasa butonul cu touch.

Acest nivel e semnificativ mai usor decat cel anterior, curba dificultati merge in jos, din nou contrar asteptarilor. Acest nivel are si un mic easter-egg pentru exploratori. Atunci cand se arunca jos se uita in spate.

**Nivelul 7**

Ultimul nivel prezinta o complexitate aparenta, jucatorului . Dar daca continuam tematica de solutii simple si aceast nivel o are. 

Camera are o multime mare de butoane si „Tap Here” si „Collide Here”, aceste butoane modifica panoul colorat din spatele incaperii. Si cubul din nivelul 3 isi face aparitia.

Scopul acestei incaperi ca si celelalte de fapt, este de distruge- re a asteptarilor. Panoul colorat din spate isi modifica culorile la intamplare cum jucatorul apasa pe butoane. Acest lucru poate crea o tendinta de a gasi sistemul. Cum panoul din spate pare foarte important si intrigant jucatorul va fi atras de el si va rata solutia adevarata. Panoul nu face nimic altceva in afara de a-si schimba culoarea, iar cubul doar apasa butoanele „Collide Here”.

Solutia acestui nivel este foarte simpla. Trebuie doar sa continue sa mearga in usa. Aceasta se va deschide cand jucatorul intra in ea.

Acesta e un nivel mai usor si un final dezamagitor dar potrivit pentru **Paradox**. O incapere cu aparent toate provocarile anterioare dar de fapt nu se intampla nimic. Eu o consider un final potrivit pentru tematica de distrugere a asteptarilor pe care le prezinta **Paradox**.

**Programare**

Probabil aspectul care poate fi imbunatatit cel mai mult din tot proiectul este programarea. Din cauza lipsei de experienta, a trebuit sa fac foarte multe „taieri de colturi” si literale cu colturile incaperi, si figurativ cu solutii suboptime pentru multe mecanici.

Mecanica principala a jocului (miscarea) a fost luata una deja testata si eficienta, apoi modificata pentru cerintele proiectului. Intoarcerea camerei prin swipe a fost creata in totalitate de mine, nici o solutie existenta nu functiona sau nu era cea ce doream.

public void OnPointerDown (PointerEventData ped)

{

initialPos = ped.position; //initierea pozitiei de tap

OnDrag(ped);

}

public void OnDrag (PointerEventData ped)

{ //aceasta prima parte e o evidenta a “taieri de colt”, ea doar se asigura ca personajul sa nu depaseasca un unghi

Dummy.transform.eulerAngles = camControl.transform.eulerAngles;

Dummy.transform.Rotate((initialPos.y - ped.position.y) \* -1 \* Time.deltaTime \* speed, 0f, 0f);

if ((Dummy.transform.eulerAngles.x >= 0 && Dummy.transform.eulerAngles.x <= 80) || (Dummy.transform.eulerAngles.x >= 280 && Dummy.transform.eulerAngles.x <= 360))

camControl.transform.Rotate((initialPos.y - ped.position.y) \* -1 \* Time.deltaTime \* speed, 0f, 0f);

else if (Dummy.transform.eulerAngles.x < 280 && Dummy.transform.eulerAngles.x > 270)

Dummy.transform.eulerAngles = new Vector3(280, 0f, 0f);

else

Dummy.transform.eulerAngles = new Vector3(80, 0f, 0f);

//pana aici este pentru prevenirea depasiri unghiului

player.transform.Rotate(0f, (initialPos.x - ped.position.x ) \* Time.deltaTime \* speed, 0f); //miscarea actuala a camerei

initialPos = ped.position; //reintializare pentru urmatorul frame}}

Sistemul de detectare a capacitati unui jucator este foarte banal. Dupa un anumit numar de secunde el va schimba instructiunile. Instructiunile se schimba doar cand jucatorul nu se uita. Si acest sistem a fost creat in totalitate de mine.

private void OnBecameVisible() //pentru a nu schimba instructiunile cand se uita

{

mat.mainTexture = texture[difficulty]; //schimbarea textului

}

IEnumerator countdown()

{

while (difficulty > 0)

{

Debug.Log(difficulty);

yield return new WaitForSeconds(seconds); //asteptarea numarului de secunde

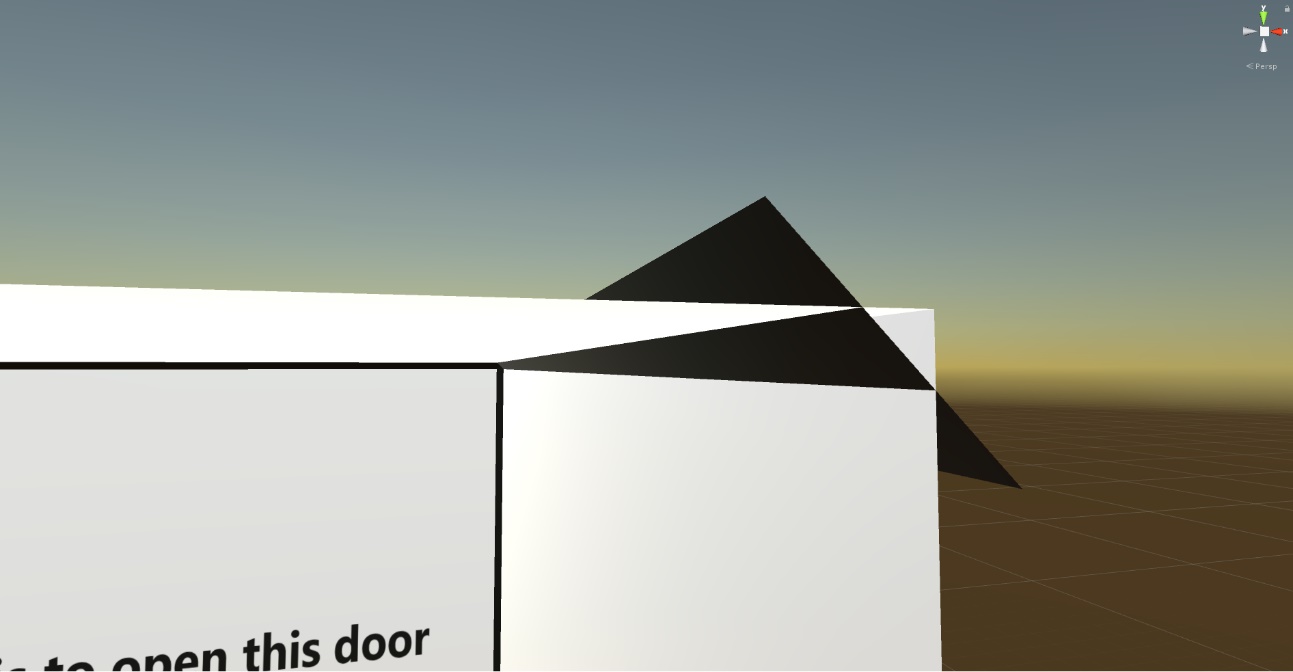
difficulty--; //reducerea dificultati de la un numar specificat pana la 1 inclusiv

}

}

Sistemul chiar daca e rudimentar e foarte eficient, el facand teste doar cand este necesar.

Efectul de colt inegrit umbrit prezent in toate camerele nu a fost facut prin folosirea unui shader (solutia optima), ci prin adaugarea unor gameObjects in colturi pentru a le inegerii. Solutia mea este suboptima dar are un efect foarte puternic si placut, adaugand si putina adancime peretilor.



Cu toate ca am lucrat singur pe intreg proiectul, am incercat sa-mi separ si cele mai simple actiuni. Cum ar fi salvatul si loadul nivelului actual, si deschiderea usii au fost tratate in propriile lor scripturi distincte de restul, ele fiind doar functii globa

le de apelat.

public void Open () //deschide usa

{

plane.color = Color.green;

anim.SetBool("State", true);

}

public void Close () //inchide usa

{

plane.color = Color.red;

anim.SetBool("State", false);

}

Iar pentru save/load am folosit:

public static void Save (string level)

{

BinaryFormatter bf = new BinaryFormatter();

FileStream file = File.Create(Application.persistentDataPath + "/savedGames.youlilhacker"); //savedGames.youlilhackerlhacker a fost ales doar pentru comedie

bf.Serialize(file ,level );

file.Close();

}

public static void Load ()

{

if (File.Exists(Application.persistentDataPath + "/savedGames.youlilhacker"))

{

BinaryFormatter bf = new BinaryFormatter();

FileStream file = File.Open(Application.persistentDataPath + "/savedGames.youlilhacker",FileMode.Open);

SaveLoad.Level = (string)bf.Deserialize(file);

}}

Ambele scripturi au fost create de mine.

O limitare foarte mare care din nou putea fi evitata cu cunoasterea shaderurilor este textul, pentru a nu putea fi vazut prin pereti Text din Unity nu s-a putut folosi. De acea eu am optat pentru o solutie mult mai suboptima. Fiecare text este un gameObject caruia i s-a alocat un material unic cu textul unic scris si salvat ca .png. Pentru ocazile cand textul se schimba am folosit un singur material si i-am schimbat doar mainTexture specific.

Cu toate limitarile mele consider ca progamarea a mers foarte bine, si ca scripturile mele sunt suficient de eficiente pentru proiectul acesta mic.