SkyRuゲームとしていろいろがありました

まずコードの部分：

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class PlayerMovement : MonoBehaviour{

public SerialHandler serialHandler;

private bool turnLeft, turnRight;

public float speed = 10.0f;

private CharacterController myCharacterController;

bool jump = false;

bool roll = false;

private Animator myAnimator;

// Start is called before the first frame update

void Start()

{

myCharacterController = GetComponent<CharacterController>();

myAnimator = GetComponent<Animator>();

}

// Update is called once per frame

void Update()

{

turnLeft = Input.GetKeyDown(KeyCode.A);

turnRight = Input.GetKeyDown(KeyCode.D);

jump = Input.GetKeyDown(KeyCode.Space);

roll = Input.GetKeyDown(KeyCode.S);

if (turnLeft)

transform.Rotate(new Vector3(0f, -90f, 0f));

else if (turnRight)

transform.Rotate(new Vector3(0f, 90f, 0f));

else if (jump)

myAnimator.Play("Jump");

else if (roll)

myAnimator.Play("Roll");

myCharacterController.SimpleMove(new Vector3(0f, 0f, 0f));

myCharacterController.Move(transform.forward \* speed \* Time.deltaTime);

}

}

**修改过后：**

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class PlayerMovement : MonoBehaviour

{

public SerialHandler serialHandler;

private bool turnLeft, turnRight;

public float speed = 10.0f;

private CharacterController myCharacterController;

bool jump = false;

bool roll = false;

private Animator myAnimator;

// Start is called before the first frame update

void Start()

{

myCharacterController = GetComponent<CharacterController>();

myAnimator = GetComponent<Animator>();

}

// Update is called once per frame

void Update()

{

// WASDを矢印に修正する

turnLeft = Input.GetKeyDown(KeyCode.LeftArrow);

turnRight = Input.GetKeyDown(KeyCode.RightArrow);

jump = Input.GetKeyDown(KeyCode.Space);

roll = Input.GetKeyDown(KeyCode.DownArrow); // 将S键更改为DownArrow

if (turnLeft)

transform.Rotate(new Vector3(0f, -90f, 0f));

else if (turnRight)

transform.Rotate(new Vector3(0f, 90f, 0f));

else if (jump)

myAnimator.Play("Jump");

else if (roll)

myAnimator.Play("Roll");

// 前進

myCharacterController.SimpleMove(new Vector3(0f, 0f, 0f));

myCharacterController.Move(transform.forward \* speed \* Time.deltaTime);

}

}

**Arduino IDE （4つ圧力センサー）**

#include <SoftwareSerial.h>

SoftwareSerial mySerial(0, 1); // ソフトウェアシリアルポートの設定（ピン0とピン1）

void setup() {

Serial.begin(9600); // ハードウェアシリアル通信の設定

mySerial.begin(9600); // ソフトウェアシリアル通信の設定

while (!Serial) {

; // シリアルポートが利用可能になるまで待機

}

Serial.println("Ready");

}

void loop() {

int sensorValue1 = analogRead(A0); // A0のアナログセンサーの値を読み取る

int sensorValue2 = analogRead(A1); // A1のアナログセンサーの値を読み取る

int sensorValue3 = analogRead(A2); // A2のアナログセンサーの値を読み取る

int sensorValue4 = analogRead(A3); // A3のアナログセンサーの値を読み取る

// シリアルモニターに値を表示

Serial.print(sensorValue1);

Serial.print(",");

Serial.print(sensorValue2);

Serial.print(",");

Serial.print(sensorValue3);

Serial.print(",");

Serial.println(sensorValue4);

// ソフトウェアシリアルポートに値を送信

mySerial.print(sensorValue1);

mySerial.print(",");

mySerial.print(sensorValue2);

mySerial.print(",");

mySerial.print(sensorValue3);

mySerial.print(",");

mySerial.println(sensorValue4);

delay(100); // 少しの待機時間を設ける

}

要在您提供的Arduino代码中添加Unity通信部分，您需要在loop()函数中添加代码以发送传感器数据到Unity。在您的代码中，传感器数据通过软串口mySerial发送到Arduino的端口。以下是如何修改您的Arduino代码

#include <SoftwareSerial.h>

SoftwareSerial mySerial(0, 1); // ソフトウェアシリアルポートの設定（ピン0とピン1）

void setup() {

Serial.begin(9600); // ハードウェアシリアル通信の設定

mySerial.begin(9600); // ソフトウェアシリアル通信の設定

while (!Serial) {

; // シリアルポートが利用可能になるまで待機

}

Serial.println("Ready");

}

void loop() {

int sensorValue1 = analogRead(A0); // A0のアナログセンサーの値を読み取る

int sensorValue2 = analogRead(A1); // A1のアナログセンサーの値を読み取る

int sensorValue3 = analogRead(A2); // A2のアナログセンサーの値を読み取る

int sensorValue4 = analogRead(A3); // A3のアナログセンサーの値を読み取る

// シリアルモニターに値を表示

Serial.print(sensorValue1);

Serial.print(",");

Serial.print(sensorValue2);

Serial.print(",");

Serial.print(sensorValue3);

Serial.print(",");

Serial.println(sensorValue4);

// ソフトウェアシリアルポートに値を送信

mySerial.print(sensorValue1);

mySerial.print(",");

mySerial.print(sensorValue2);

mySerial.print(",");

mySerial.print(sensorValue3);

mySerial.print(",");

mySerial.println(sensorValue4);

// 在这里添加代码以发送数据到Unity（可以使用Serial.println()或Serial.write()等方法）

delay(100); // 少しの待機時間を設ける

}

**Arduino と　Unity 相互通信んを行う場合：**

**Arudino側：**

const int sensorPins[] = {A0, A1, A2, A3}; // 圧力センサーのアナログ入力ピン

void setup() {

Serial.begin(9600); // シリアル通信の初期化

}

void loop() {

for (int i = 0; i < 4; i++) {

int sensorValue = analogRead(sensorPins[i]); // 各圧力センサーから値を読み取る

Serial.print(sensorValue);

Serial.print(",");

}

Serial.println(); // 改行を送信

delay(100); // 適宜遅延を入れる

}

**Unity側：**

using UnityEngine;

using System.IO.Ports;

public class ArduinoCommunication : MonoBehaviour

{

SerialPort serialPort;

int[] sensorValues = new int[4];

void Start()

{

serialPort = new SerialPort("COM3", 9600); // COMポートとボーレートを指定

serialPort.Open(); // シリアルポートをオープン

// 受信イベントの設定

serialPort.DataReceived += new SerialDataReceivedEventHandler(OnDataReceived);

}

void OnDataReceived(object sender, SerialDataReceivedEventArgs e)

{

string data = serialPort.ReadLine(); // データを読み取る

string[] values = data.Split(',');

if (values.Length == 4)

{

for (int i = 0; i < 4; i++)

{

sensorValues[i] = int.Parse(values[i]);

}

}

}

void Update()

{

// sensorValues配列を使って各センサーのデータを処理

// 例: sensorValues[0]はセンサー1の値、sensorValues[1]はセンサー2の値、依次類推

}

void OnDestroy()

{

if (serialPort != null && serialPort.IsOpen)

{

serialPort.Close(); // シリアルポートをクローズ

}

}

}

**Game Timer Set: 30s - 0s**

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.UI;

public class CountDown : MonoBehaviour

{

GameObject timerText;

float time = 30.0f;

// Start is called before the first frame update

void Start()

{

this.timerText = GameObject.Find("Time");

}

// Update is called once per frame

void Update()

{

this.time -= Time.deltaTime;

if(time > 0.0f)

{

this.timerText.GetComponent<Text>().text = this.time.ToString("F1");

}

else

{

this.time = 0.0f;

this.timerText.GetComponent<Text>().text = this.time.ToString("F1");

}

}

}

**Game timer set 2分間**

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.UI;

public class CountDown : MonoBehaviour

{

GameObject timerText;

float time = 60.0f; // タイムを1分間に変更

// Start is called before the first frame update

void Start()

{

this.timerText = GameObject.Find("Time");

}

// Update is called once per frame

void Update()

{

this.time -= Time.deltaTime;

if (time > 0.0f)

{

this.timerText.GetComponent<Text>().text = this.time.ToString("F1");

}

else

{

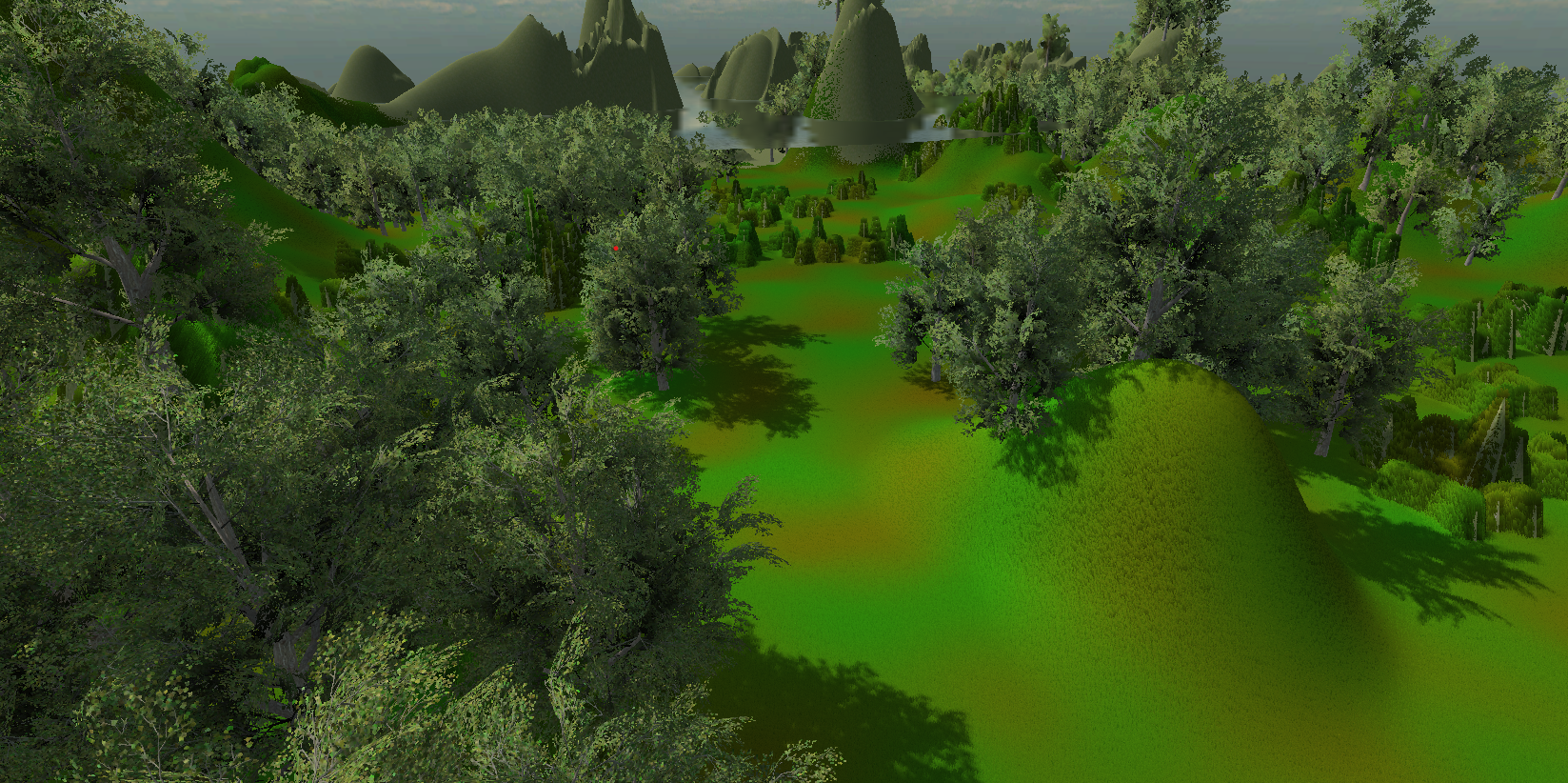
this.time = 0.0f;

this.timerText.GetComponent<Text>().text = this.time.ToString("F1");

}

}

}



インタフェース：

Arduinoプログラムの作成:

int fsrPins[] = {A0, A1, A2, A3}; // 各FSRセンサーのアナログピン

int fsrReadings[4] = {0}; // 各センサーの読み取り値

void setup() {

Serial.begin(9600);

}

void loop() {

for (int i = 0; i < 4; i++) {

fsrReadings[i] = analogRead(fsrPins[i]);

Serial.print(fsrReadings[i]); // センサーの生のアナログ読み取り値を送信

Serial.print(","); // カンマを区切り文字として使用

// 各センサーの圧力レベルを識別するための条件分岐を追加

if (fsrReadings[i] < 1) {

Serial.println("0"); // "no pressure"

} else if (fsrReadings[i] > 3) {

Serial.println("1"); // "light touch"

} else if (fsrReadings[i] > 500) {

Serial.println("2"); // "light squeeze"

} else if (fsrReadings[i] > 800) {

Serial.println("3"); // "medium squeeze"

} else {

Serial.println("4"); // "big squeeze"

}

}

delay(100);

}

Unityプロジェクトの設定:

Arduinoからのデータを受信するために、UnityにSerialPortを使用するライブラリが必要です。このライブラリをUnityプロジェクトにインポートします。

Unityスクリプトの作成:

Unity内でArduinoからのデータを受信し、データを扱うスクリプトを作成します。以下はUnityのC#スクリプトの一部です。

using System.Collections;

using System.Collections.Generic;

using System.IO.Ports;

using UnityEngine;

public class vw : MonoBehaviour

{

SerialPort Stream = new SerialPort("COM6", 9600);

int fsrReading;

void Start()

{

Stream.Open();

}

void Update()

{

if (Stream.IsOpen)

{

try

{

string h = Stream.ReadLine();

fsrReading = int.Parse(h);

}

catch (System.Exception)

{

// Handle parsing or communication errors

}

}

}

}

