Chicken Robot Simulation
치킨 제조 협동로봇 시뮬레이션게임

201620244 신문혁

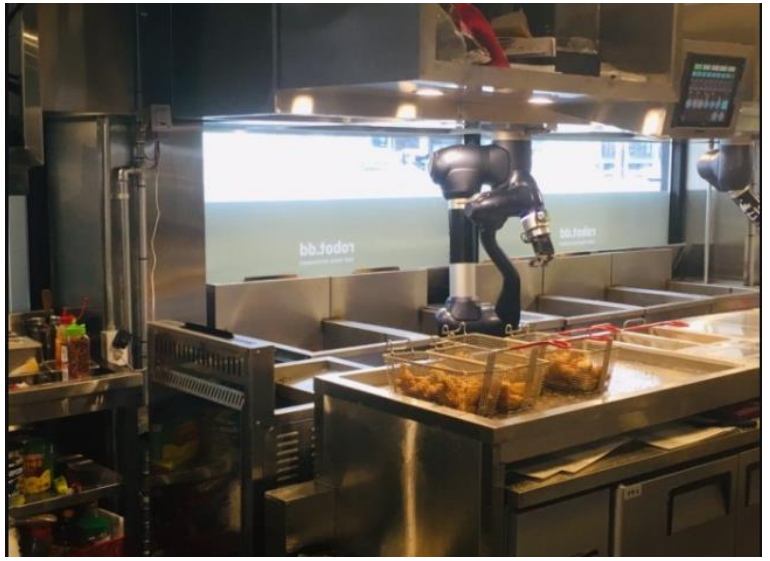
목 차

1. 서 론

1.1 주제선정 배경

산업용 로봇

산업용 로봇 시장



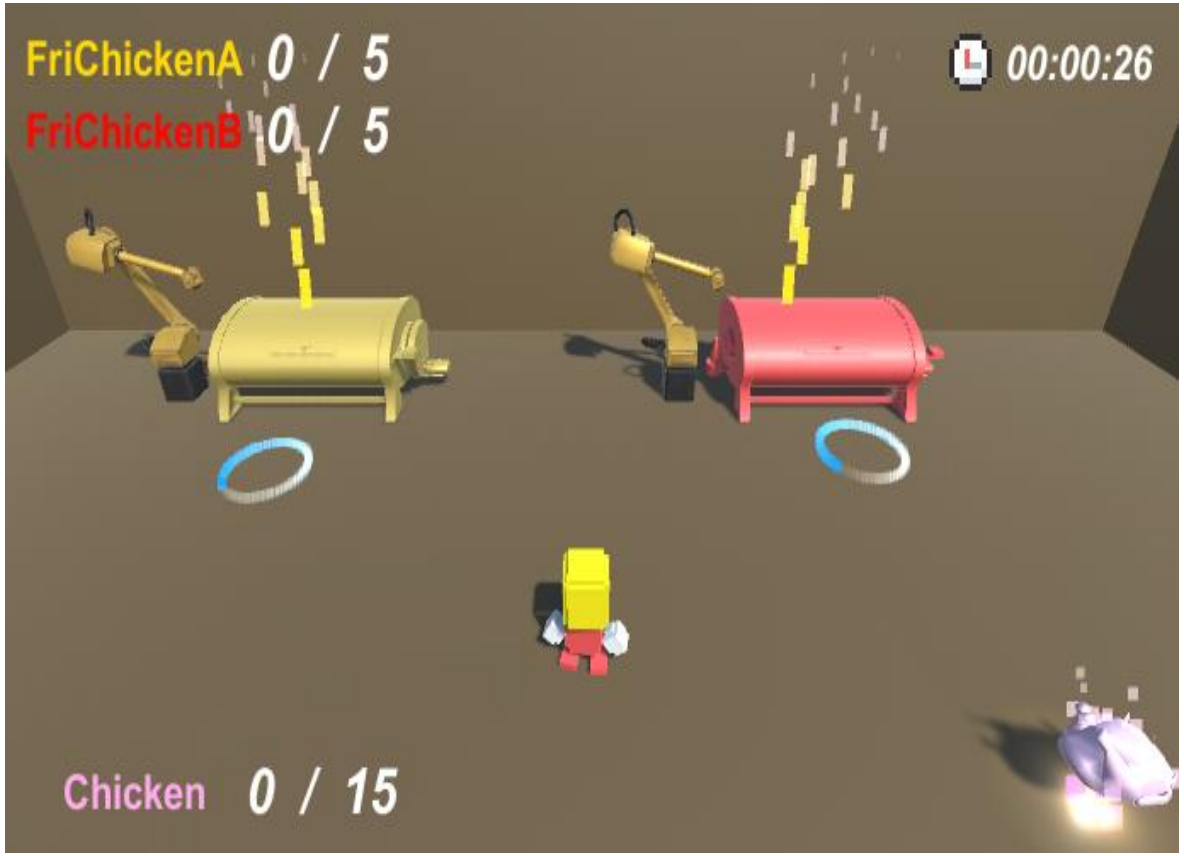
1. 서 론

1.2 문제정의 및 상황인식

Chicken Robot Simulation
치킨 제조 협동로봇
시뮬레이션게임

2. 본 론

2.1 시뮬레이션 모델 설계



Chicken Robot Simulation

- 캐릭터 설정
- 도구 설정
- 도구를 이용하여 협동로봇 작동
- 치킨 재료와 협동로봇으로 치킨 조리
- 치킨 습득

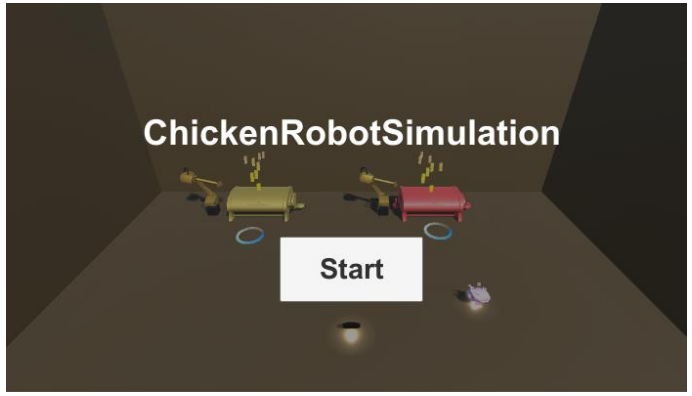
2. 본 론

2.2 C# 코드 설명

```
public class GameManager : MonoBehaviour
{
    public GameObject menuCam;
    public GameObject inGameCam;
    public Player player;
    public float playTime;

    public GameObject menuPanel;
    public GameObject gamePanel;
    public GameObject overPanel;
    public Text friChickenAText;
    public Text friChickenBText;
    public Text playTimeText;
    public Text playerChicknText;

    void Awake()
    {

    }

    public void GameStart()
    {
        menuCam.SetActive(false);
        inGameCam.SetActive(true);

        menuPanel.SetActive(false);
        gamePanel.SetActive(true);
    }
```

GameManager Class

- 게임내 object와 판넬 설정
- 게임 시작



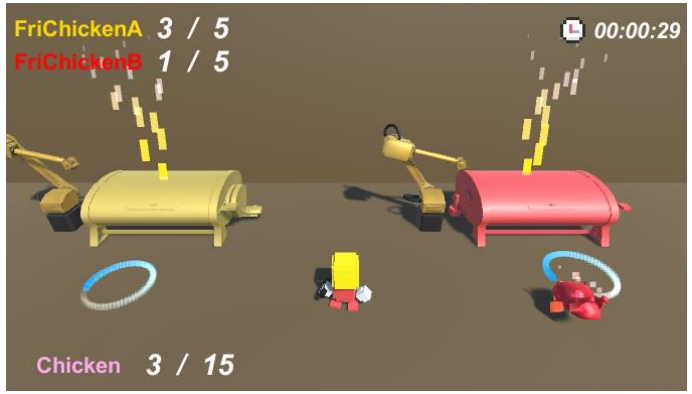
2. 본 론

2.2 C# 코드 설명

```
void Update()
{
    playTime += Time.deltaTime;
}

void LateUpdate()
{
    //시간UI
    int hour = (int)(playTime / 3600);
    int min = (int)((playTime - hour * 3600) / 60);
    int second = (int)(playTime % 60);
    playTimeText.text = string.Format("{0:00}", hour) + ":"
                      + string.Format("{0:00}", min) + ":"
                      + string.Format("{0:00}", second);

    //완성된 치킨UI
    friChickenAText.text = player.friChickenA + "  /  " + player.maxFriChickenA;
    friChickenBText.text = player.friChickenB + "  /  " + player.maxFriChickenB;

    //치킨 재료UI
    playerChicknText.text = player.chicken + "  /  " + player.maxChicken;
}

public void GameOver()
{
    gamePanel.SetActive(false);
    overPanel.SetActive(true);
}

public void Restart()
{
    SceneManager.LoadScene(0);
}
```

GameManager Class

- 게임내 object와 판넬 설정
- 게임 시작
- 게임 종료
- 종료 메뉴



2. 본 론

2.2 C# 코드 설명

```
public float speed;
public GameObject[] gloves;
public bool[] hasGloves;
public GameManager manager;

public int chicken;
public int friChickenA;
public int friChickenB;
public int maxChicken;
public int maxFriChickenA;
public int maxFriChickenB;


float hAxis;
float vAxis;

bool interDown;
bool swapDown;
bool pushDown;
bool isPushReady = true;
bool isFinish = false;

Vector3 moveVec;

Animator anim;

GameObject nearObject;
Glove equipGlove;
float pushDelay;
int equipGloveIndex = -1;
```

Player Class

- 캐릭터 정보 변수
- 캐릭터 움직임 변수
- 키와 상태 변수
- 도구 설정 변수

2. 본 론

2.2 C# 코드 설명

```
void Update()
{
    GetInput();
    Move();
    Swap();
    ButPush();
    Interaction();
}

void GetInput()
{
    hAxis = Input.GetAxisRaw("Horizontal");
    vAxis = Input.GetAxisRaw("Vertical");
    pushDown = Input.GetButtonDown("Button");
    interDown = Input.GetButtonDown("Interaction");
    swapDown = Input.GetButtonDown("Swap");
}

void Move()
{
    moveVec = new Vector3(hAxis, 0, vAxis).normalized;

    transform.position += moveVec * speed * Time.deltaTime;

    anim.SetBool("IsWalk", moveVec != Vector3.zero);

    transform.LookAt(transform.position + moveVec);
}
```

Player Class

- 함수 설정
- 키 입력 함수 설정
- 캐릭터 이동 함수 설정

2. 본 론

2.2 C# 코드 설명

```
public class Tool : MonoBehaviour
{
    public enum Type { Glove, Chicken, FriChickenA, FriChickenB };
    public Type type;
    public int value;

    void Update()
    {

    }
}
```

Tool Class

➢ 도구 설정

Player Class

➢ 재료(치킨) 줍기

```
void OnTriggerEnter(Collider other)
{
    if (other.tag == "Tool")
    {
        Tool tool = other.GetComponent<Tool>();
        switch (tool.type)
        {
            case Tool.Type.Chicken:
                chicken += tool.value;
                if (chicken > maxChicken)
                    chicken = maxChicken;
                break;
            case Tool.Type.FriChickenA:
                friChickenA += tool.value;
                if (friChickenA > maxFriChickenA)
                    friChickenA = maxFriChickenA;
                if (chicken > 0)
                    chicken -= 1;
                Destroy(other.gameObject);
                    break;
            case Tool.Type.FriChickenB:
                friChickenB += tool.value;
                if (friChickenB > maxFriChickenB)
                    friChickenB = maxFriChickenB;
                if (chicken > 0)
                    chicken -= 1;
                Destroy(other.gameObject);
                break;
        }
    }
```

2. 본 론

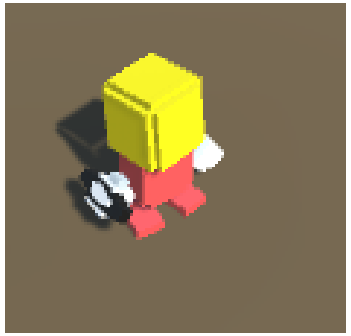
2.2 C# 코드 설명

```
void Interaction()
{
    if (interDown && nearObject != null)
    {
        if (nearObject.tag == "Glove")
        {
            Tool tool = nearObject.GetComponent<Tool>();
            int gloveIndex = tool.value;
            hasGloves[gloveIndex] = true;

            Destroy(nearObject);
        }
    }
}
```

Player Class

- 도구(장갑) 줍기
- 도구(장갑) 착용

```
void Swap()
{
    if (swapDown && !hasGloves[0])
        return;

    int gloveIndex = -1;
    if (swapDown) gloveIndex = 0;

    if (swapDown)
    {
        if (equipGlove != null)
            equipGlove.gameObject.SetActive(false);

        equipGloveIndex = gloveIndex;
        equipGlove = gloves[gloveIndex].GetComponent<Glove>();
        equipGlove.gameObject.SetActive(true);
    }

}
```



2. 본 론

2.2 C# 코드 설명

```
public class Glove : MonoBehaviour
{

    public enum Type { App };
    public Type type;
    public float rate;
    public BoxCollider pushArea;
    public TrailRenderer trailEffect;

    public void Use()
    {
        StopCoroutine("Push");
        StartCoroutine("Push");
    }

    IEnumerator Push()
    {
        yield return new WaitForSeconds(0.1f);
        pushArea.enabled = true;
        trailEffect.enabled = true;

        yield return new WaitForSeconds(0.3f);
        pushArea.enabled = false;

        yield return new WaitForSeconds(0.3f);
        trailEffect.enabled = false;
    }
}
```

```
void ButPush()
{
    if (equipGlove == null)
        return;

    pushDelay += Time.deltaTime;
    isPushReady = equipGlove.rate < pushDelay; //true

    if (pushDown && isPushReady)
    {
        equipGlove.Use();
        anim.SetTrigger("doPush");
        pushDelay = 0;
    }
}
```

Glove Class

- 변수설정
- Push Co-Routine 설정

Player Class

- ButPush 사용

2. 본 론

2.2 C# 코드 설명

```
public class Robot : MonoBehaviour
{
    public GameObject[] chickenObj;
    public Transform[] chickenPos;

    Rigidbody rigid;
    BoxCollider boxCollider;
    Material mat;
    Player worker;

    void Awake()
    {
        rigid = GetComponent<Rigidbody>();
        boxCollider = GetComponent<BoxCollider>();
        mat = GetComponent<MeshRenderer>().material;
    }

    IEnumerator OnPush()
    {
        mat.color = Color.red;
        yield return new WaitForSeconds(0.1f);
        mat.color = Color.white;
    }
```

Robot Class

- 변수설정
- OnPush Co-Routine 설정

2. 본 론

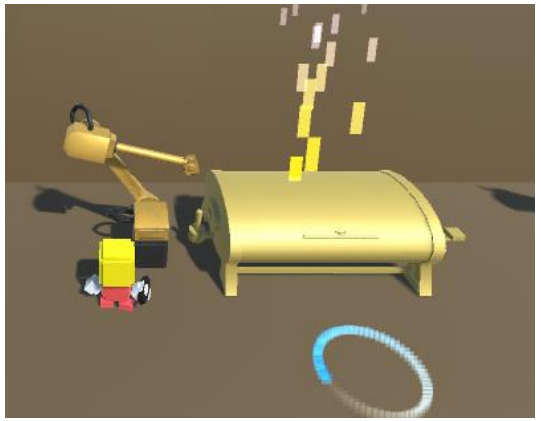
2.2 C# 코드 설명

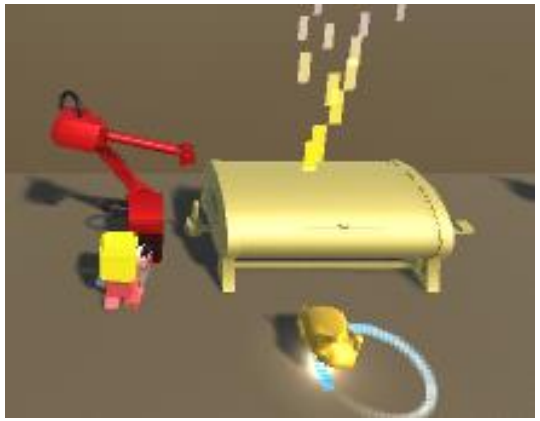
```
public void Cook()
{
        Vector3 ranVec = Vector3.right * Random.Range(-3, 3)
            + Vector3.forward * Random.Range(-3, 3);
        Instantiate(chickenObj[0], chickenPos[0].position + ranVec, chickenPos[0].rotation);
}
```

Robot Class

- Cook 함수설정
- 함수 사용

```
public void OnTriggerEnter(Collider other)
{
    if (other.tag == "App")
    {
        StartCoroutine(OnPush());
        Cook();
    }
}
```





2. 본 론

2.2 C# 코드 설명

```
if (friChickenA == maxFriChickenA && friChickenB == maxFriChickenB && !isFinish)
{
    OnFinish();
}
```

```
void OnFinish()
{
    isFinish = true;
    manager.GameOver();
}
```

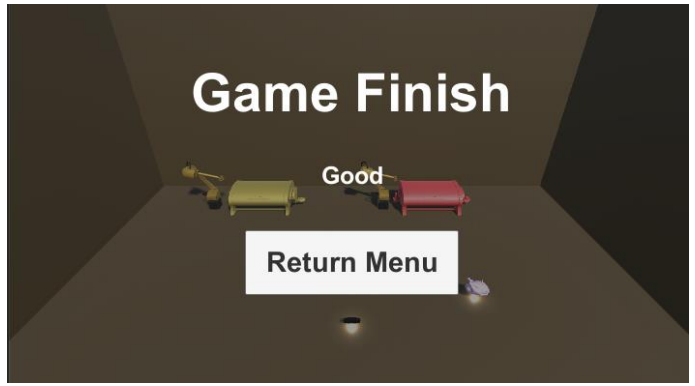
```
public void GameOver()
{
    gamePanel.SetActive(false);
    overPanel.SetActive(true);
}
```

Player Class

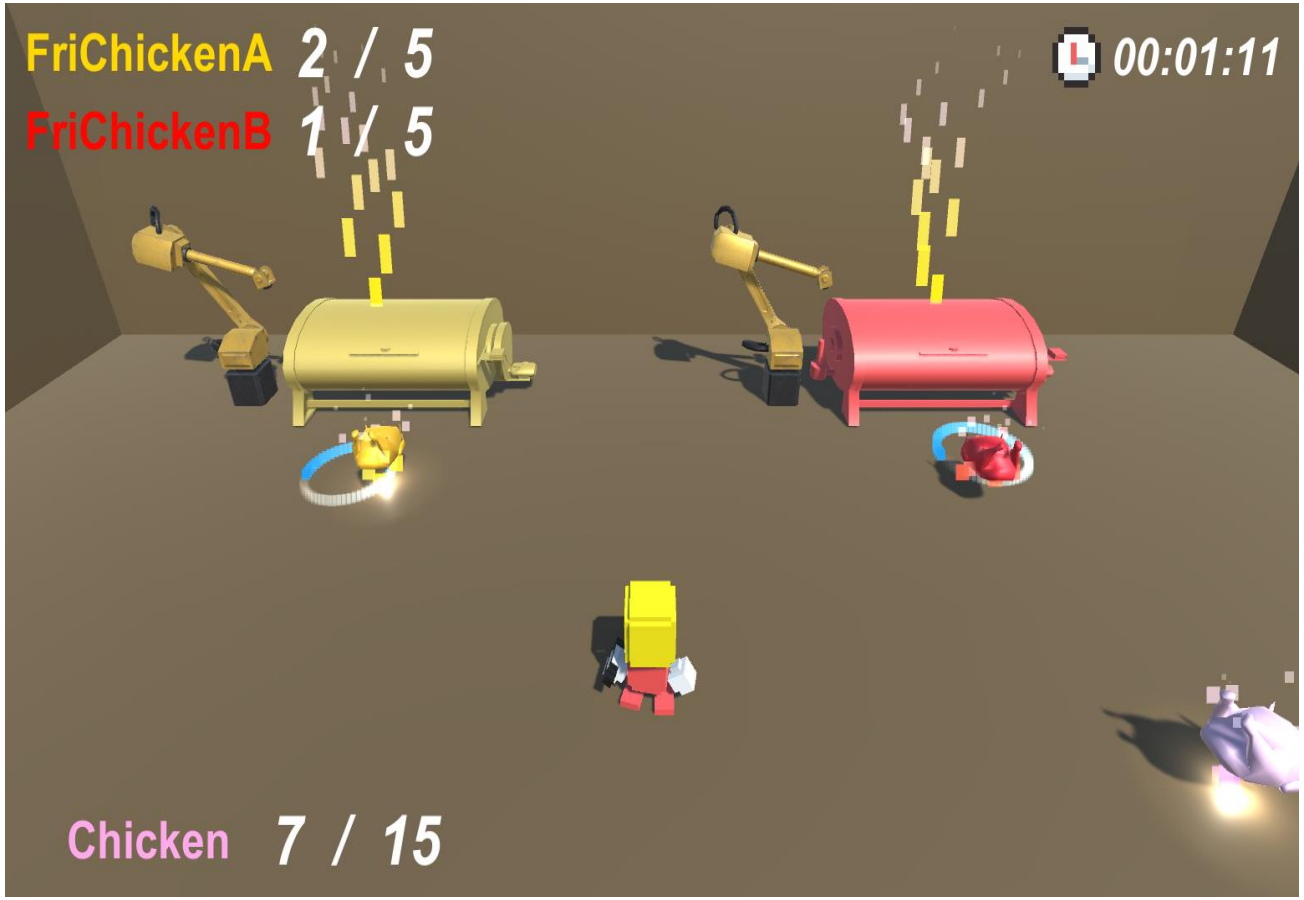
- 종료조건
- 종료 함수

GameManager Class

- 게임 종료 함수



3. 결 론




```
void Awake()
        ⬇
public void GameStart()
        ⬇
void Update()
        ⬇
void LateUpdate()
        ⬇
public void GameOver()
```

감사합니다.