NPC Controller

Dmytro Symovonyk, Vsevolod Kharchenko

JOHANNES KEPLER
UNIVERSITY LINZ

Asynchronous Programming solution

```
Unity Message | 0 references
private async void Start()
{
    await SpawnNPCsAsync();
}

1 reference
private async Task SpawnNPCsAsync()
{
    int total = teamSize * 2;
    npcTransforms = new TransformAccessArray(total);
    positions = new NativeArray<Vector3>(total, Allocator.Persistent);
    hp = new NativeArray<int>(total, Allocator.Persistent);
    team = new NativeArray<int>(total, Allocator.Persistent);
    lastAttackTime = new NativeArray<float>(total, Allocator.Persistent);
    attackResults = new NativeArray<AttackResult>(total, Allocator.Persistent);
    animationStates = new NativeArray<AnimationState>(total, Allocator.Persistent);

    for (int i = 0; i < total; i++)
    {
        Vector3 pos = new Vector3(Random.Range(-spawnRange, spawnRange), 0, Random.Range(-spawnRange, spawnRange));
        GameObject prefab = i < teamSize ? npcTeam1 : npcTeam2;
        GameObject npc = Instantiate(prefab, pos, Quaternion.identity);
        npc.name = $"NPC_{i}";
        npcTransforms.Add(npc.transform);
        positions[i] = pos;
        hp[i] = 100;
        team[i] = (i < teamSize) ? 0 : 1;
        lastAttackTime[i] = -attackCooldown;
        attackResults[i] = new AttackResult { targetIndex = -1, damage = 0 };
        animationStates[i] = AnimationState.Moving;

        if (i % 50 == 0)
            await Task.Yield();
    }
    spawned = true;
}
```

- Problem: instanciating all NPCs at once.
- Here we used async to load NPCs before the game starts, in slices, in order to run the game itself more smoothly afterwards.

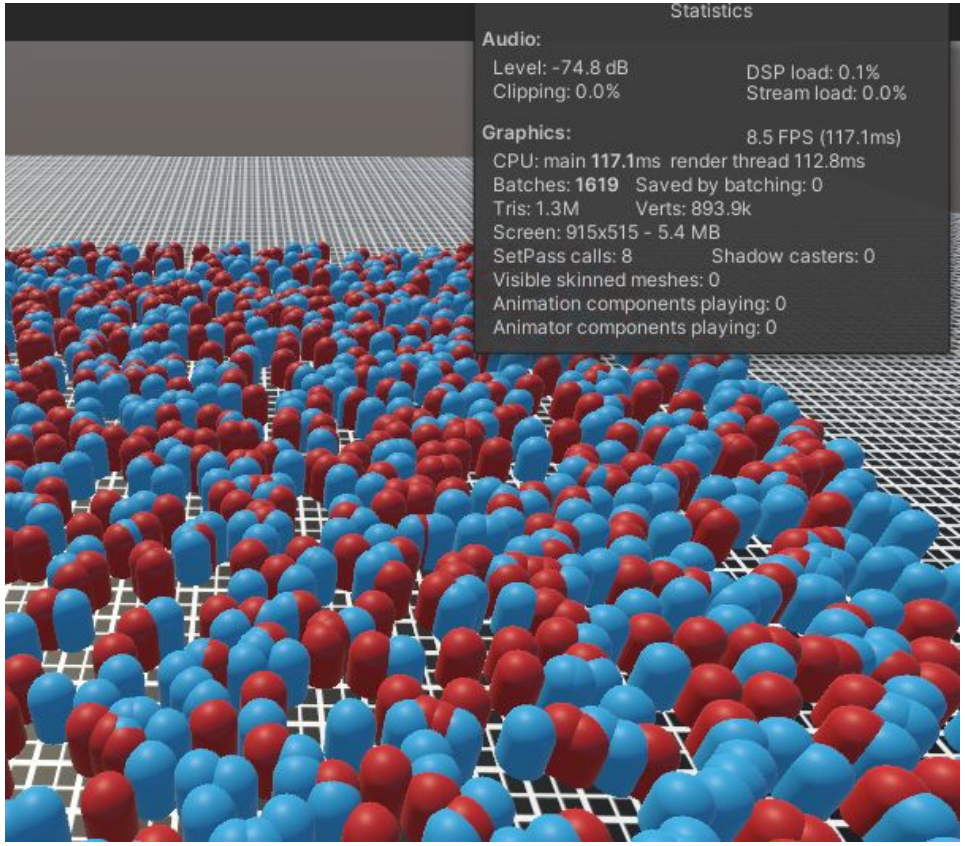
Parallel Programming Solution

```
[BurstCompile]
1 reference
struct NPCLogicJob : IJobParallelForTransform
{
    public float deltaTime;
    public float time;
    public float moveSpeed;
    public float attackRange;
    public float attackDamage;
    public float attackCooldown;

    [ReadOnly] public NativeArray<Vector3> positions;
    [ReadOnly] public NativeArray<int> team;
    [ReadOnly] public NativeArray<int> hp;
    public NativeArray<float> lastAttackTime;
    public NativeArray<AttackResult> attackResults;
    public NativeArray<AnimationState> animationStates;

    0 references
    public void Execute(int index, TransformAccess transform)...
```

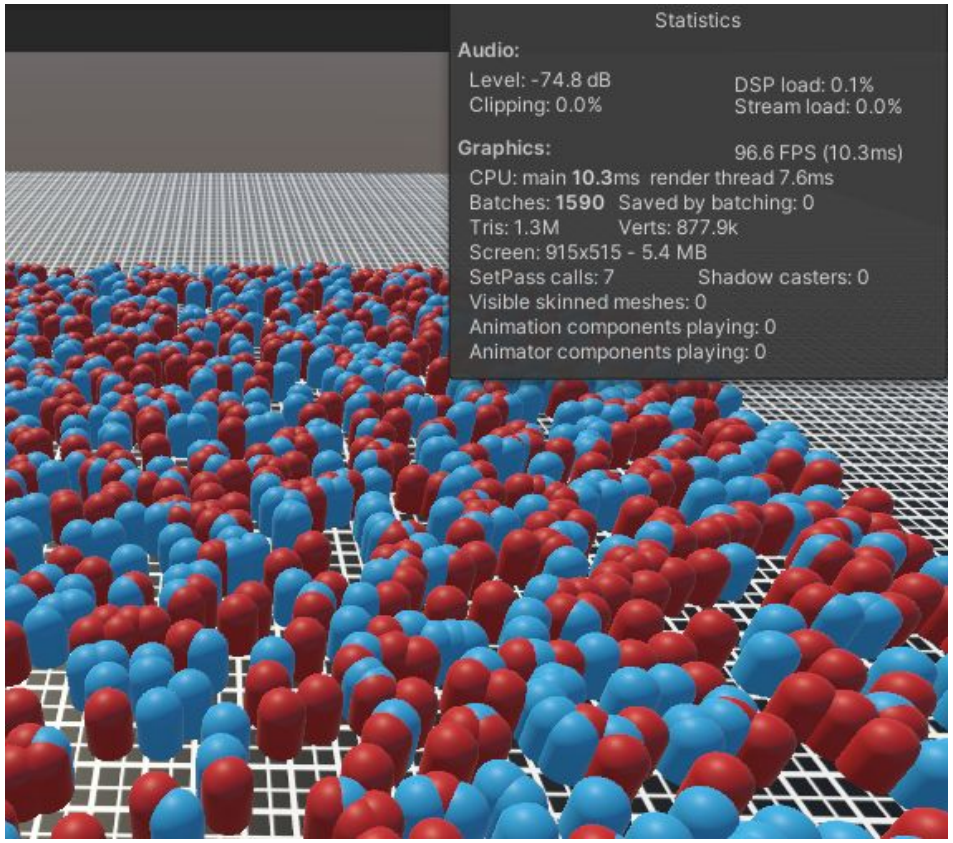
- Problem: heavy computations during runtime on Main Thread
- Parallel programming used to handle NPC logic on multiple Threads, freeing the main one

```
Unity Message | 0 references
void Update()
{
    if (!spawned || npcTransforms.length == 0)
        return;

    // creating copy of positions for read only access in the job
    var positionsCopy = new NativeArray<Vector3>(positions, Allocator.TempJob);

    var job = new NPCLogicJob
    {
        deltaTime = Time.deltaTime,
        time = Time.time,
        moveSpeed = moveSpeed,
        attackRange = attackRange,
        attackDamage = attackDamage,
        attackCooldown = attackCooldown,
        positions = positionsCopy,
        hp = hp,
        team = team,
        lastAttackTime = lastAttackTime,
        attackResults = attackResults,
        animationStates = animationStates
    };

    jobHandle = job.Schedule(npcTransforms);

    //disposes after job is complete
    jobHandle = positionsCopy.Dispose(jobHandle);
}

Unity Message | 0 references
void LateUpdate()
{
    if (!spawned || npcTransforms.length == 0)
        return;

    jobHandle.Complete();
```

Results (tested on 2000 NPCs)



No optimization



With optimization

JOHANNES KEPLER
UNIVERSITÄT LINZ

Thank you

JOHANNES KEPLER
UNIVERSITY LINZ
Altenberger Straße 69
4040 Linz, Austria
jku.at