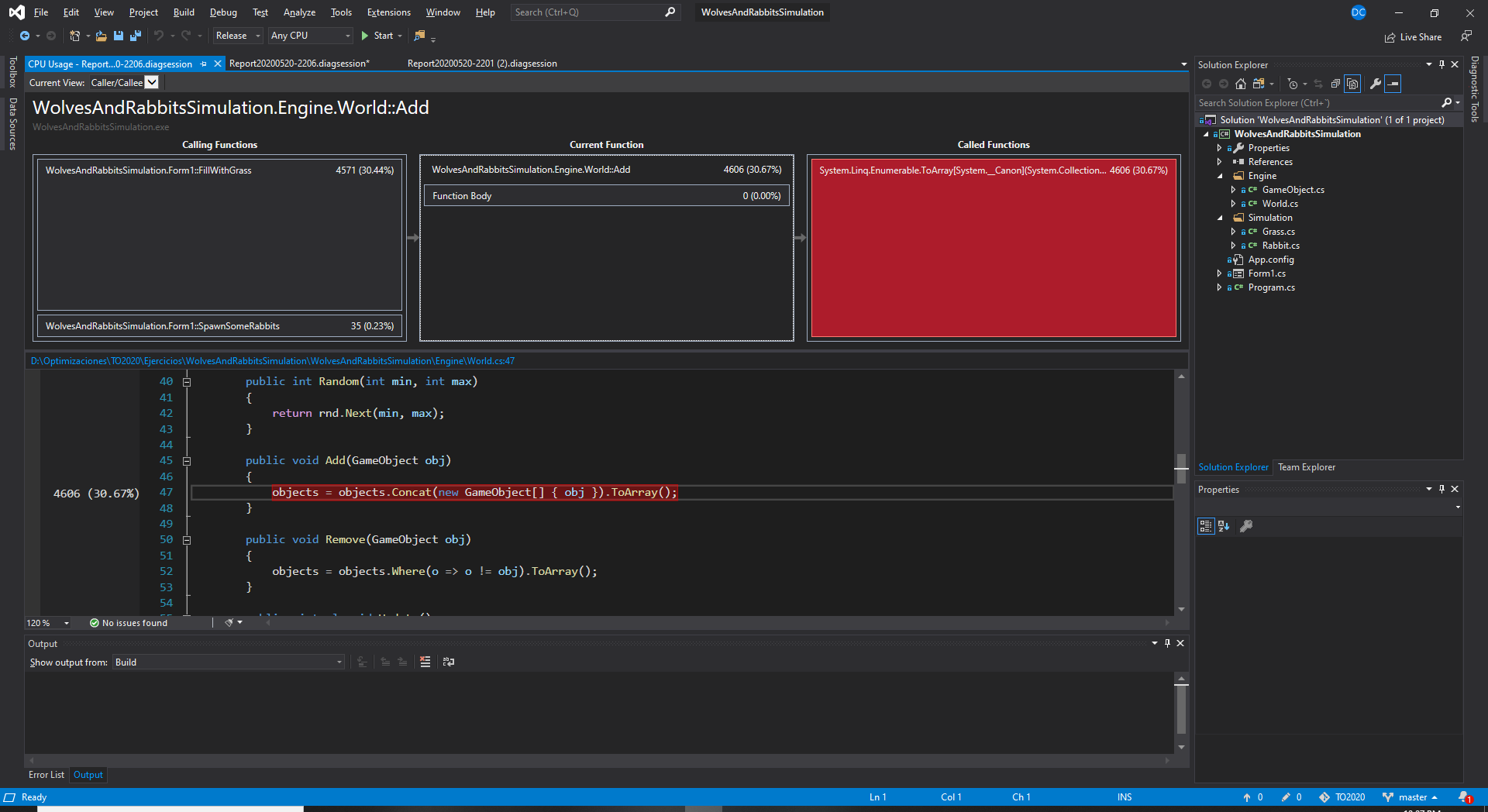
Consignas  
1. Realizar mediciones de performance que permitan diagnosticar el problema.  
2. Escribir qué problemas encontró en la simulación en función de las mediciones.  
3. Proponer una solución al problema encontrado, describirla en sus palabras con el mayor detalle posible.  
4. Implementar la solución planteada haciendo los cambios en el código que sean necesarios.  
5. Validar la implementación realizando nuevas mediciones. ¿Se resolvió el problema? En caso negativo, ¿qué otras soluciones alternativas se le ocurren?

1)

Captura de pantalla de computadora

Descripción generada automáticamente



2)

El programa funciona lento al principio, aproximadamente a 10 fps que mide el programa por default. A medida que progresa el tiempo y spawnean mas conejos el programa procede a ralentizarse cada vez mas hasta llegar a unos 3 fps donde el programa fue cerrado. A continuacion enumerare que funciones que mas CPU ocupan para luego analizar su codigo.

La función “public IEnumerable<GameObject> ObjectsAt(Point pos)” utiliza mucho CPU. Esto se debe a que recorre toda la lista de ambos conejos y pasto (objetos) cada vez que es llamada. Esto empeora más adelante mientras más conejos haya ya que deberá recorrerlo una mayor cantidad de veces, por eso los fps empeoran con el tiempo.

También la función “public void Add(GameObject obj)” utiliza mucho CPU. Creo que sucede lo mismo en este caso, al querer concatenar el array la línea “objects = objects.Concat(new GameObject[] { obj }).ToArray();” recorre denuevo todos los objetos dentro del array. Creo que optimizar estas 2 funciones ayudaría mucho a la velocidad del programa.

3)

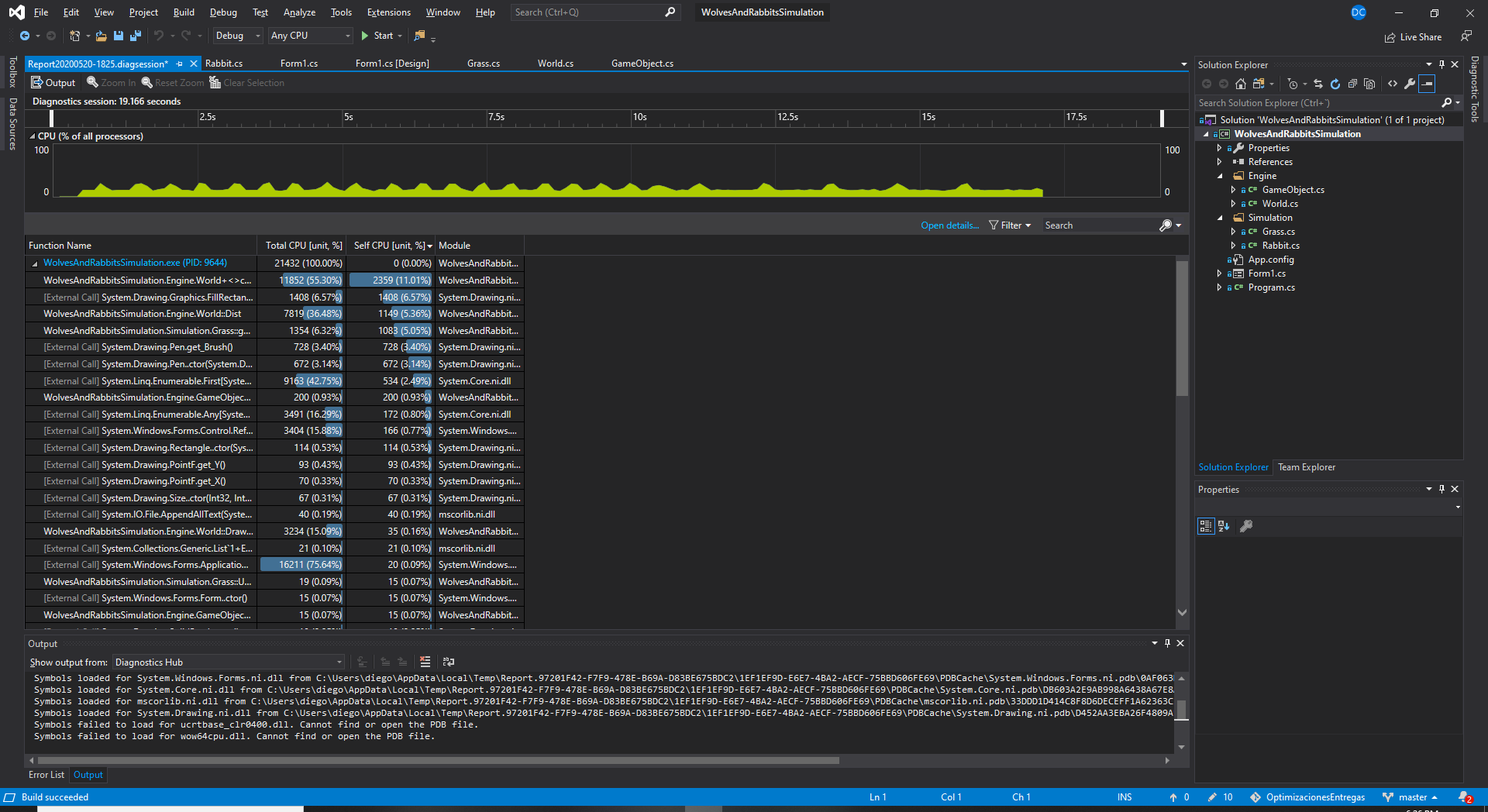
Lo que se me ocurre que podría solucionar el problema es dividir a los conejos y al pasto en 2 listas distintas, esto en cuanto al Add haría que no tenga que recorrer a todos los objetos al querer agregar uno, simplemente se haría un listaConejos.Add(conejoNuevo). En cuanto al ObjectsAt separar la lista debería mejorar un poco la performance ya que recorrería menos objetos al separar los conejos.

4)

Se tuvieron que cambiar mas funciones ya que el array GameObjects se utilizaba en varias funciones.

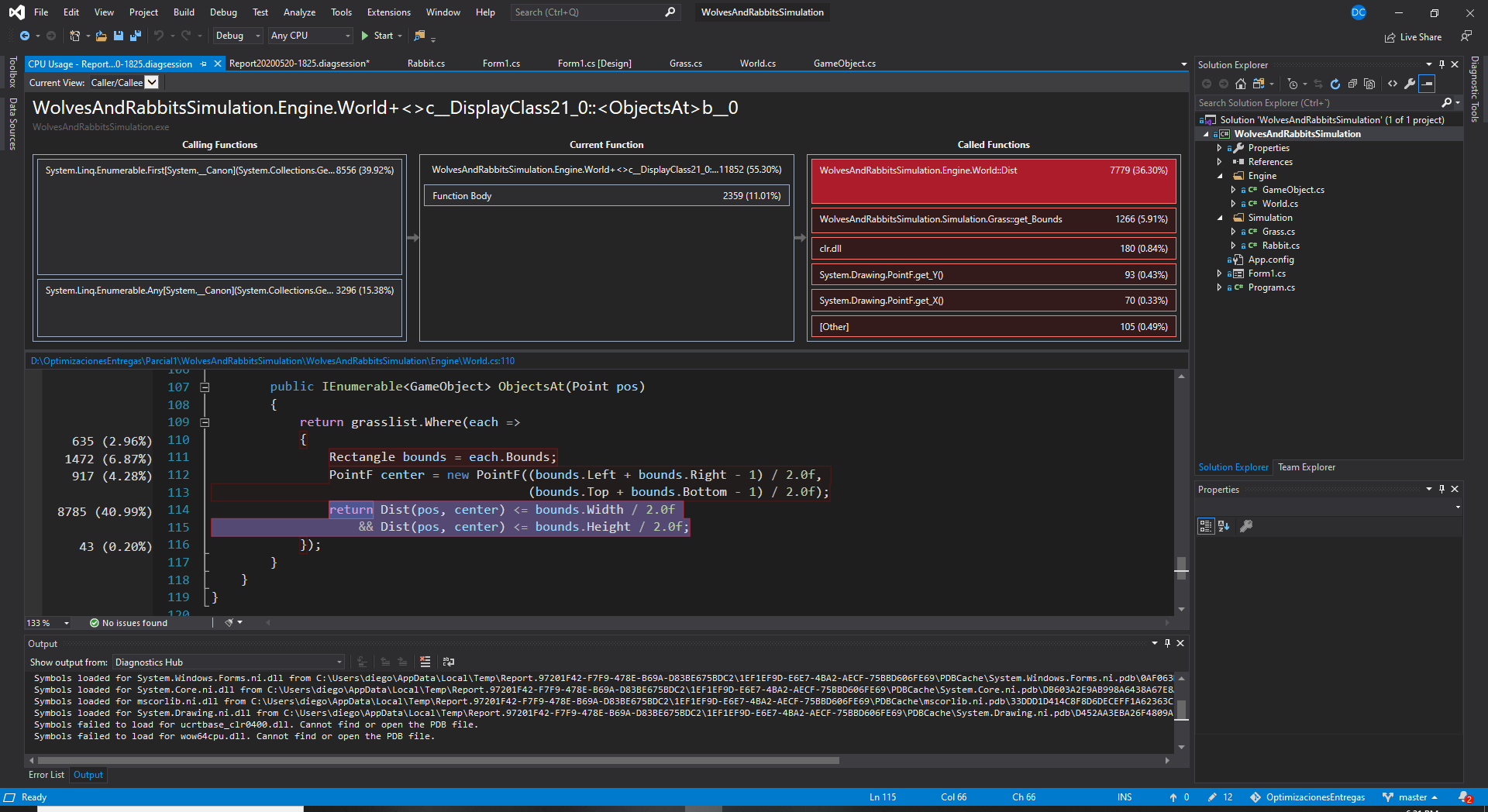
5)

Estos cambios no resolvieron el problema, los frames mejoraron poco y nada.



Para resumir el programa se ejecuta a 1 o 2 frames mas por segundo antes del frame 100 y luego cae a 4 fps en vez de a 3 fps. Analizando mas en profundidad el codigo y las primeras mediciones me di cuenta que el mayor problema esta en las lineas que retorna la funcion ObjectsAt ya que recorre todos los pastos por conejo para saber sobre que pasto esta parado y se puede comer. Para hacer esto la funcion llama a otra funcion Dist que hace calculos matematicos que estan ocupando mucha CPU. Las lineas que mas CPU consumen son : return Dist(pos, center) <= bounds.Width / 2.0f

&& Dist(pos, center) <= bounds.Height / 2.0f;



Procederé a intentar cambiarlas por completo o optimizarlas lo mejor posible en un segundo intento para ver si puedo mejorar los frames del programa.

Procedí a hacer cambios al código, la función ObjectsAt la modifiqué por completo eliminando también la función Dist. El código procedió a mejorar ya que no hace tantos cálculos como antes. Sigue sin estar optima pero paso a unos 20 fps antes del frame 100 y a una estabilidad mucho mayor en aproximadamente 13 fps despues del frame 100.

