게임알고리즘기초과정

과 목 명 : 게임일반프로그래밍

능력단위 : C/C++ 게임알고리즘

제출일자 : 2023년 08월 07일

포트폴리오 : Octree가 적용된 지형에서 정적 및 동적 개체의 제외 및 이동 처리 구현

작성자 : 오시진

<제출내역>

1. Octree 개체 이동 및 제외 처리 프로그램
2. 프로그램 구현 분석 및 세부 문서

<CONTENTS>

1. 프로젝트 소개 및 개요
   1. 프로젝트 소개
   2. 프로젝트 주요 기술
2. 프로젝트 설계 및 다이어그램
   1. 시퀀스 다이어그램
   2. 클래스 다이어그램
      1. 프로젝트에서 사용된 변수 역할
      2. 프로젝트에서 사용된 함수 역할
3. 최종 결과 및 추가 내용
   1. 범용성/유연성/확장성/간결성 고려하여 추가된 내용
      1. 현재 프로젝트 설계(구현 내용)
      2. 향후 개발 내용
   2. 최종 결과

1. 프로젝트 소개 및 개요
   1. 프로젝트 소개

대상 프로젝트는 3차원 공간 분할(Octree)이 적용된 지형에서 정적 및 동적 개체의 이동 및 제외 처리를 구현한다. 자료구조 중 map, set 등을 사용하여 Object를 관리한다. 오브젝트는 정적(Static), 동적(Dynamic) 오브젝트로 나누어져 있고 데이터 또한 포인터로 관리하기 때문에 동적 할당 등 메모리 관리 측면에서 프로그램을 설계한다. 또한 확장성과 재사용성을 위해 기능별로 클래스를 구분하여 모듈화한다. 그리고 개체의 이동과 충돌 등 벡터 계산을 위한 기능을 직접 구현하였다.

* 1. 프로젝트 주요 기술
     1. 3차원 공간 분할(Octree)

3차원 공간을 8진트리로 분할하여 개체의 충돌을 검사할 때 같은 Subspace 내에 있는 오브젝트만 확인하기 때문에 연산이 효율적이다. 또한 어떤 공간 노드에 어떤 오브젝트가 포함되어 있는지 각 노드의 멤버 컨테이너를 통해 관리하는 것으로 공간 분할의 장점을 극대화했다.

* + 1. 개체의 이동 및 충돌 처리
       1. 개체는 중심점(pivot)과 부피(Volume)를 지니는 3차원 직육면체로 화면 좌표계 내에 존재한다. 개체는 또한 Target 벡터 방향으로 계속해서 이동하면서 다른 개체와의 충돌 여부를 검사하고 만약 충돌이 발생했다면 두 개체 모두 공간에서 제외한다.

1. 프로젝트 설계 및 다이어그램
   1. 시퀀스 다이어그램

텍스트, 도표, 평행, 평면도이(가) 표시된 사진

자동 생성된 설명텍스트, 스크린샷, 번호, 폰트이(가) 표시된 사진

자동 생성된 설명텍스트, 스크린샷, 평행, 번호이(가) 표시된 사진

자동 생성된 설명텍스트, 도표, 영수증, 번호이(가) 표시된 사진

자동 생성된 설명

* 1. 클래스 다이어그램

텍스트, 스크린샷, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

* + 1. 프로젝트에서 사용된 변수 역할

|  |  |  |
| --- | --- | --- |
| 클래스명 | 변수 | 내용 |
| Objcet | iListIndex | ObjectList의 인덱스 변수 |
| iNodeIndex | 공간 노드의 인덱스 변수 |
| mBox | 위치와 부피를 보관하는 변수 |
| DynamicObject | fSpeed | 오브젝트의 속도에 곱해지는 속력 변수 |
| mDirection | 오브젝트의 방향 단위 벡터 변수 |
| mTarget | 오브젝트의 목적지를 설정하는 변수 |
| Box | mbEnable | 개체의 사용 여부를 나타내는 변수 |
| mCenter | 개체의 중심 벡터를 저장하는 변수 |
| mHalf | 개체의 너비, 높이, 깊이의 절반을 저장하는 변수 |
| mMax | 개체의 정점 중 가장 큰 좌표를 저장하는 변수 |
| mMin | 개체의 정점 중 가장 작은 좌표를 저장하는 변수 |
| mPoint | 개체의 정점들을 저장하는 변수 |
| size | 개체의 너비, 높이, 깊이를 저장하는 변수 |
| V | 개체의 Pivot 위치를 저장하는 변수 |
| Vector3 | X | 개체의 x좌표를 저장하는 변수 |
| y | 개체의 y좌표를 저장하는 변수 |
| Z | 개체의 z좌표를 저장하는 변수 |
| Octree | iNumCounter | Octree의 total subspace를 저장하는 변수 |
| mNodeList\_DynObj | 동적 오브젝트가 포함되어 있는 공간 노드를 보관하는 변수 |
| mNodeList\_StaObj | 정적 오브젝트가 포함되어 있는 공간 노드를 보관하는 변수 |
| mpRootNode | 최상위 부모 공간 노드 변수 |
| NodeIndex | 각 노드들을 순서대로 보관하는 std::vector 변수 |
| Queue | 순회를 위해 사용되는 std::queue 변수 |
| Node | mBox | 공간 노드의 영역을 보관하는 변수 |
| mDynamicObjectList | 노드에 포함된 동적 오브젝트들을 보관하는 변수 |
| mStaticObjectList | 노드에 포함된 정적 오브젝트들을 보관하는 변수 |
| miDepth | 해당 노드의 8진트리에서의 깊이를 보관하는 변수 |
| miIndex | 공간 노드의 index를 저장하는 변수 |
| mpChild | 노드의 자식 노드들을 보관하는 std::vector 변수 |
| mpParent | 노드의 부모 노드를 보관하는 포인터형 변수 |
| Timer | mdwBeForeTime | 프레임 간 시간차이를 계산하기 위해 기록해 놓은 시간 변수 |
| mfAccumulatedSecond | mfSPF의 누적합을 보관하는 변수 |
| mfGameTimer | 게임이 실행된 후 총 누적시간을 나타내는 변수 |
| mfSPF | Second Per Frame을 저장하는 변수 |
| miFPS | Frame Per Second를 저장하는 변수 |

* + 1. 프로젝트에서 사용된 함수 역할

|  |  |  |
| --- | --- | --- |
| 클래스명 | 함수 | 내용 |
| Object | Move | Object를 이동 |
| SetBox | Object의 영역을 설정 |
| SetTarget | Object의 목표 위치를 설정 |
| Box | BoxToBox | 개체 간 충돌을 검사 |
| ToBox | 나와 다른 개체 간 충돌 검사 |
| Vector3 | Angle | 두 벡터 간 사이각을 산출 |
| GetDistance | 두 벡터 간 거리를 산출 |
| Length | 벡터의 크기를 산출 |
| Normalize | 벡터의 정규화 |
| NormVector | 단위벡터 산출 |
| Octree | AddDynamicObject | 노드에 포함된 동적 오브젝트 추가 |
| AddStaticObject | 노드에 포함된 정적 오브젝트 추가 |
| BuildTree | 트리 생성 |
| CheckCollision | 충돌 검사 |
| CreateNode | 노드 생성 |
| FindNode | 노드 검색 |
| Frame | 매 프레임 시 실행되는 함수 |
| Init | 초기화 |
| LevelOrder | 같은 깊이의 노드 순으로 순회 |
| PostFrame | 매 프레임 직후 실행 |
| PreFrame | 매 프레임 직전 실행 |
| PreOrder | 먼저 방문하는 노드 순으로 순회 |
| Release | 트리 종료 시 실행 |
| Render | 트리 정보를 출력 |
| Run | Octree를 실행 |
| Node | SetParent | 부모 노드 설정 |
| Timer | Frame | 매 프레임 실행 |
| GetFPS | FPS를 산출 |
| Init | 초기화 |
| Release | 클래스 종료 |
| Render | 시간 정보를 화면에 출력 |

1. 최종 결과 및 추가 내용
   1. 범용성/유연성/확장성/간결성 고려하여 추가된 내용
      1. 현재 프로젝트 설계(구현 내용)
         1. 3차원 공간을 8진트리로 분할하여 개체의 이동 및 개체 간 충돌과 렌더링 최적화
         2. Target 방향으로 동일한 속도로 이동하는 동적 오브젝트
         3. 이동 중 다른 오브젝트와 충돌 여부를 검사 가능
         4. 벡터 간 다양한 연산을 지원. 내적과 외적 및 사이각을 구하며 거리와 프로젝션 등이 가능.
      2. 향후 개발 내용
         1. 개체의 선형 이동 뿐만 아니라 Scaling, Rotate, Transform 등의 기능 구현
         2. 벡터의 계산 뿐만 아니라 행렬을 사용한 병렬 연산으로 프로그램 최적화
         3. 난수 생성 장치를 개선하여 프로그램 실행 시 항상 동일하지 않은 난수로 개체가 이동하게끔 변경
         4. 코드에 불필요한 반복문을 줄이고 적절한 예외처리 등을 통해 프로그램의 안전성과 유지보수성을 증가
         5. 충돌 후 개체가 사라짐에 따라 일정 시간 후 모든 개체가 사라지는 것을 보완하여 충돌로 사라진 개체들이 다시 공간에 스폰되게끔 설정
   2. 최종 결과 화면

