고급 소프트웨어 실습(CSE4152)

1주차

Monte Carlo Simulation

목차

- 실험 환경
- Monte Carlo simulation
- Random number generation (난수 생성)
- 실습
- 과제

실험환경

- Window Visual Studio 2019
- C++ 의 Win32 programming
- 공개된 Win32 SDK 를 이용
- Visual Studio 설치

Monte Carlo Simulation

- 확정 모형(deterministic model)이 아닌 확률 모형(stochastic model)에서는 분석적인 방법으로 해를 찾을 수 없음
- 몬테카를로 시뮬레이션(Monte Carlo simulation)은 랜덤 샘플링 기법을 반복하여 시뮬레이션을 수행하고, 원하는 수치적 결과를 전체 확률 분포에서 계산해내는 계산 알고리즘의 한 종류
- 몬테카를로 시뮬레이션은 수치 적분, 확률 분포 계산, 확률 기반 최적화 등의 분야에서 사용

- Simulation: 동전 던지기를 모사하기 위하여 [0,1] 범위에서 random value를 추출하여 0.5 미만을 tail로, 0.5 이상을 head로 간주
- Monte Carlo method: 동전을 한 박스 쏟아서 head와 tail의 숫자를 세고, 이로부터 head가 나올 확률을 구함
- Monte Carlo simulation: [0,1] 범위에서 random value를 반복해서 추출, 이로부터 head를 얻을 확률을 구함

Monte Carlo Simulation

- Monte Carlo simulation의 일반적인 패턴은 다음과 같다.

1. 샘플링을 진행할 영역 정의
2. 정의한 영역에 대한 랜덤 샘플링
3. 수집한 샘플들에 대해 결정론적 계산(deterministic computation) 수행
4. 결과를 집계하여 근사치 도출

Monte Carlo Simulation

- 몬테카를로 방법의 간단한 예시로 주사위를 두 번 던져 8의 합이 나오는 경우의 확률을 구해볼 수 있음

1. [1,6] 범위의 난수를 두 개 생성하여, 주사위를 두 번 던지는 시행, 100번을 랜덤하게 실행하는 시뮬레이션을 진행
2. 두 난수의 합이 8이면 hit, 아닐 경우 miss라고 가정하고 hit와 miss의 횟수 집계
3. 전체 시행 횟수에 대한 hit의 비율 계산
4. 수학적으로 계산한 확률과 비교

Monte Carlo Simulation

- 몬테카를로 방법의 또 다른 예시로 원주율을 구하는 것을 들 수 있음

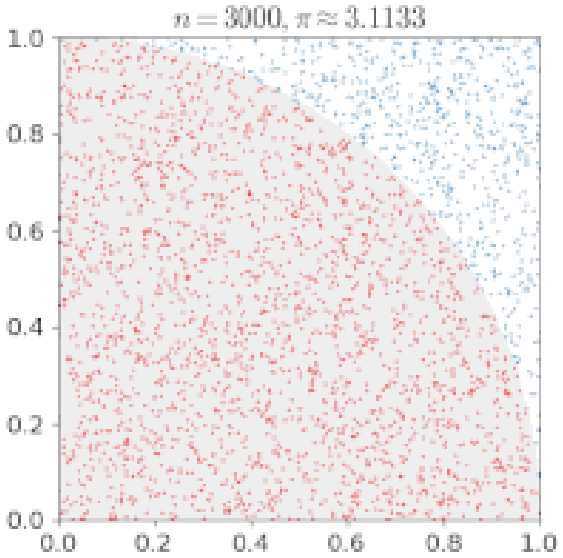


그림 1. 몬테카를로 방법으로 원주율을 계산하는 과정
(https://commons.wikimedia.org/wiki/File:Pi_30K.gif#/media/파일:Pi_30K.gif)

Monte Carlo Simulation

- $x^2 + y^2 = 1$ 이라는 식으로 표현되는 원을 이용하여 원주율 계산

- 이 원은 $-1 \leq x \leq 1, -1 \leq y \leq 1$ 로 표현되는 넓이가 4인 정사각형 공간 안에 포함됨
- 이 공간 안에서 n개의 난수 순서쌍 (x, y)을 추출
- 추출한 점 중 원 내부의 점의 개수 집계
- 원 내부의 개수와 전체 개수의 비율은 두 영역의 비율은 나타냄. 이를 이용하여 π를 구함

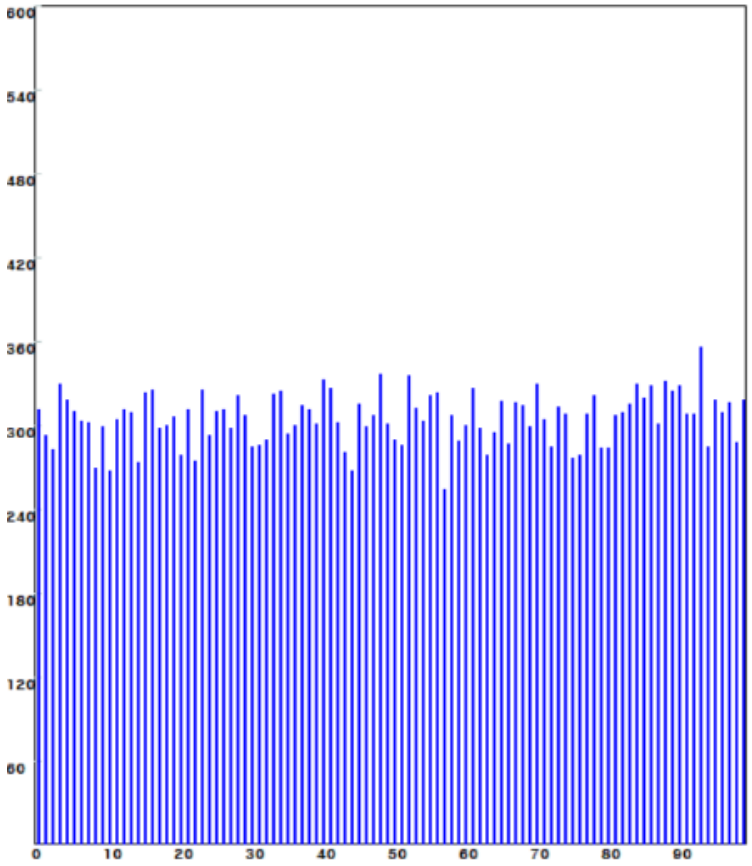
rand() 함수의 한계

- Seed 값이 너무 천천히 변하여 같은 시간대에서는 모두 같은 의사 난수 (pseudo random number) 수열을 생성
- 주어진 범위에서 균등하게 난수를 생성하지 않음 (0 ~ 32767 (RAND_MAX))
- C의 rand() 함수가 사용하는 선형합동법(Linear congruential generator) 방식의 알고리즘은 생성하는 난수 수열의 상관 관계가 높아서 비효율적임
- C++의 <random> 라이브러리, 메르센 트위스터 난수 생성기 (Mersenne Twister)

실습1. 히스토그램 그리기

- [0,99] 범위에서 stdlib.h의 rand() 사용하여 난수 생성
- 난수를 생성하는 과정을 여러 번 반복하고, 각각의 경우에서 생성된 난수에 대한 분포를 확인
- WIN32를 사용하여 분포를 히스토그램 형식으로 나타내고, 분포가 uniform distribution의 형태를 보이는지 확인

실습1. 결과 예시



실행 결과

실습 2. 주사위 던지기

- 실습은 Visual Studio 환경에서 C++의 win32 programming 을 통해 진행
- rand() 를 이용하여 MAX_TRY의 값만큼 주사위를 던짐
- 이때 첫번째 던진 주사위의 눈의 값과 두번째로 던진 주사위의 눈의 값의 합이 8인 경우를 hit으로 간주
- 몬테카를로 시뮬레이션을 사용하여 두 번 주사위를 던질 때 눈의 합이 총 8이 될 확률을 구하고 오차율을 확인

실습2. 결과 예시

try 1: 1,5	try 21: 6,6	try 41: 2,3	try 61: 3,4	try 81: 1,1
try 2: 6,2 (hit!)	try 22: 1,4	try 42: 5,4	try 62: 6,5	try 82: 5,6
try 3: 1,3	try 23: 2,6 (hit!)	try 43: 5,3 (hit!)	try 63: 2,3	try 83: 6,1
try 4: 4,3	try 24: 6,6	try 44: 5,3 (hit!)	try 64: 2,4	try 84: 4,4 (hit!)
try 5: 2,2	try 25: 3,6	try 45: 4,3	try 65: 2,5	try 85: 3,5 (hit!)
try 6: 4,2	try 26: 3,5 (hit!)	try 46: 6,1	try 66: 4,2	try 86: 1,5
try 7: 3,3	try 27: 6,1	try 47: 6,5	try 67: 6,3	try 87: 1,1
try 8: 2,1	try 28: 5,6	try 48: 5,6	try 68: 4,4 (hit!)	try 88: 6,4
try 9: 2,6 (hit!)	try 29: 6,1	try 49: 3,5 (hit!)	try 69: 1,5	try 89: 5,1
try 10: 3,1	try 30: 3,2	try 50: 2,2	try 70: 5,2	try 90: 2,5
try 11: 3,1	try 31: 4,5	try 51: 4,3	try 71: 2,6 (hit!)	try 91: 3,5 (hit!)
try 12: 6,5	try 32: 5,5	try 52: 6,6	try 72: 2,5	try 92: 2,3
try 13: 2,6 (hit!)	try 33: 2,4	try 53: 6,1	try 73: 4,5	try 93: 6,4
try 14: 2,2	try 34: 1,6	try 54: 1,2	try 74: 5,5	try 94: 6,6
try 15: 4,6	try 35: 6,6	try 55: 3,4	try 75: 3,6	try 95: 1,5
try 16: 4,1	try 36: 2,5	try 56: 2,3	try 76: 6,2 (hit!)	try 96: 1,3
try 17: 5,5	try 37: 4,4 (hit!)	try 57: 5,3 (hit!)	try 77: 2,2	try 97: 5,6
try 18: 1,6	try 38: 2,6 (hit!)	try 58: 3,6	try 78: 4,2	try 98: 1,3
try 19: 5,1	try 39: 2,4	try 59: 4,1	try 79: 3,6	try 99: 6,3
try 20: 2,3	try 40: 2,6 (hit!)	try 60: 3,2	try 80: 5,5	try 100: 5,3 (hit!)

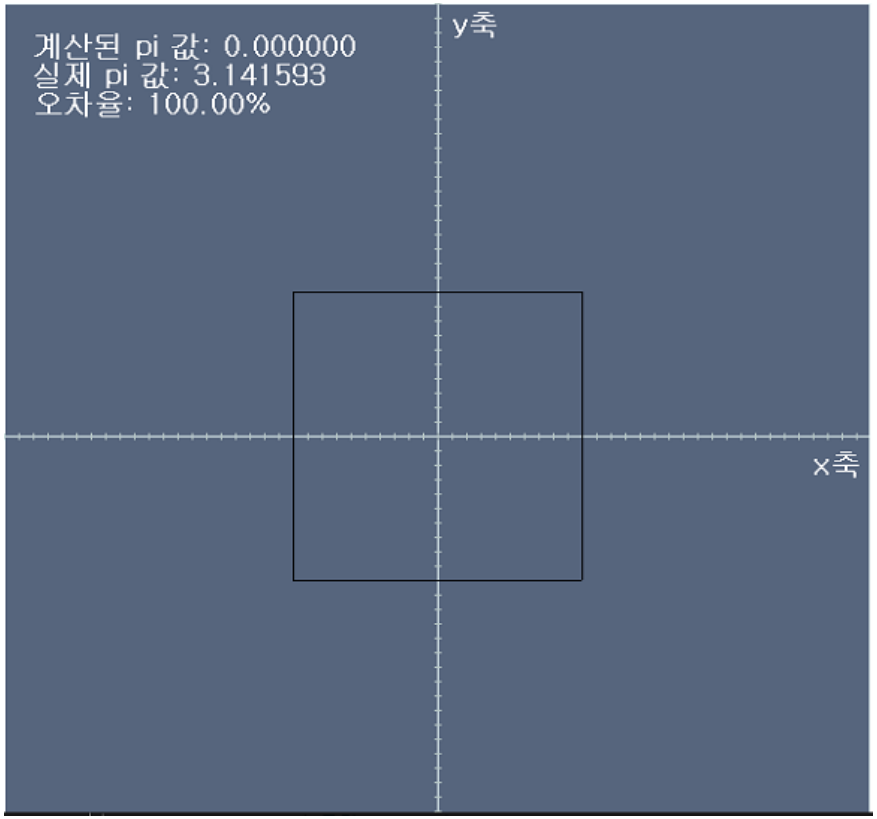
실제 값: **0.138889**
계산된 값: **0.190000**

실습 파일 결과 예시

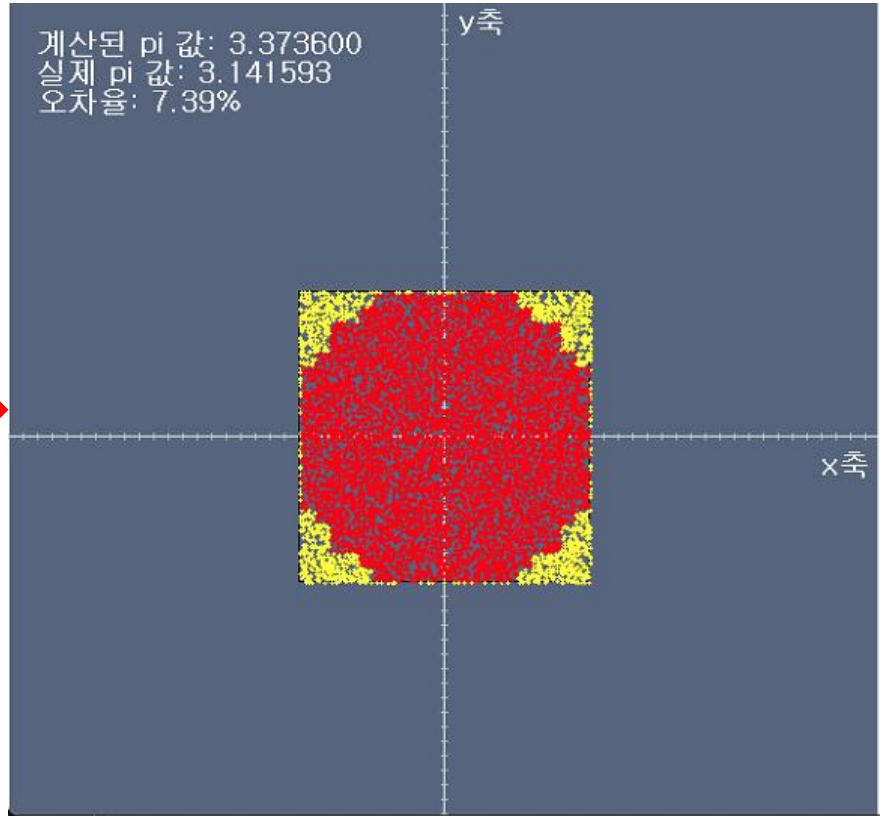
실습3. 원주율 구하기

- rand() 를 이용하여 MAX_POINTS 개수만큼 난수 순서쌍 생성을 하여 주어진 정사각형 안에 점을 그림
- 이때 주어진 정사각형 안에 접하는 원에 순서쌍이 속할 경우와 아닐 경우를 색으로 구별
- 몬테카를로 시뮬레이션을 통하여 π 를 구하고 오차율을 확인

실습3. 결과 예시



실습 파일 실행 예시



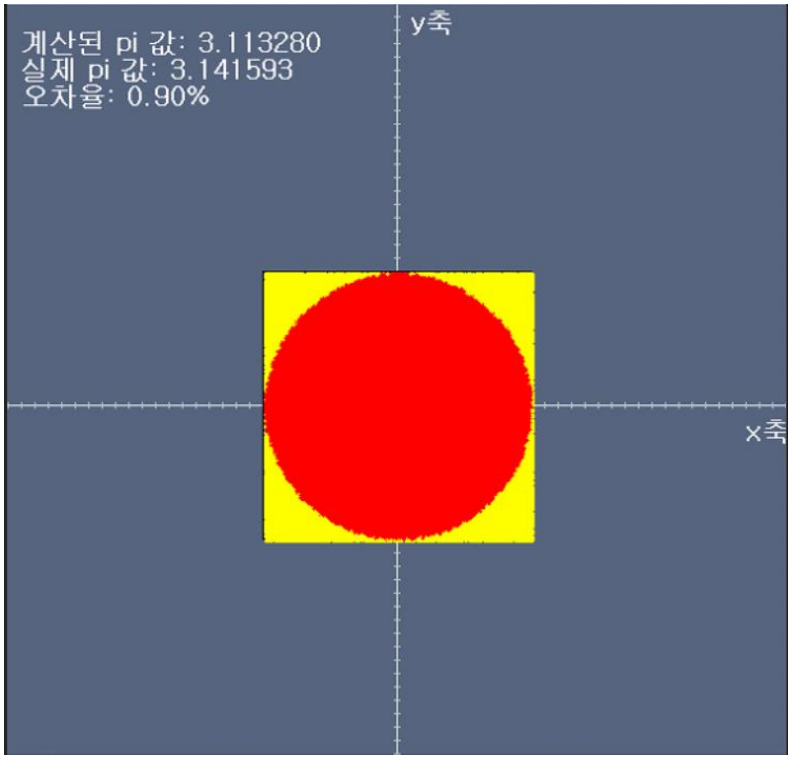
실습 파일 결과 예시

과제: Monte Carlo simulation의 오차율 개선

- 실습에서 계산한 π 의 오차율이 1% 미만이 되도록 코드 수정
- rand() 함수가 아닌 C++의 <random> 라이브러리를 이용할 것

- Linear congruential, 메르센 트위스터 등의 난수 생성 방법에 대하여 간단히 설명할 것

과제1. 결과 예시



과제결과 예시