유니티로 배우는 게임 수학

원의 표현

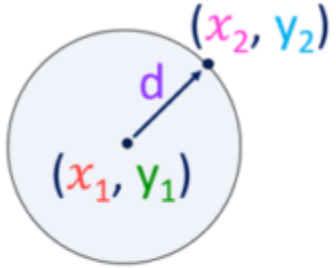
아주대학교 미디어학과

오규환 (drghoh@ajou.ac.kr)

원 : $x^2 + y^2 = r^2$

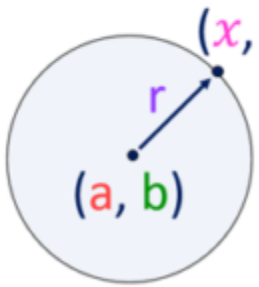
Deriving the Equation of a Circle

1. Start with the Pythagorean equation for distance between 2 points



$$\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} = d$$

2. Substitute in the values of (a, b) as the centre and r as the radius



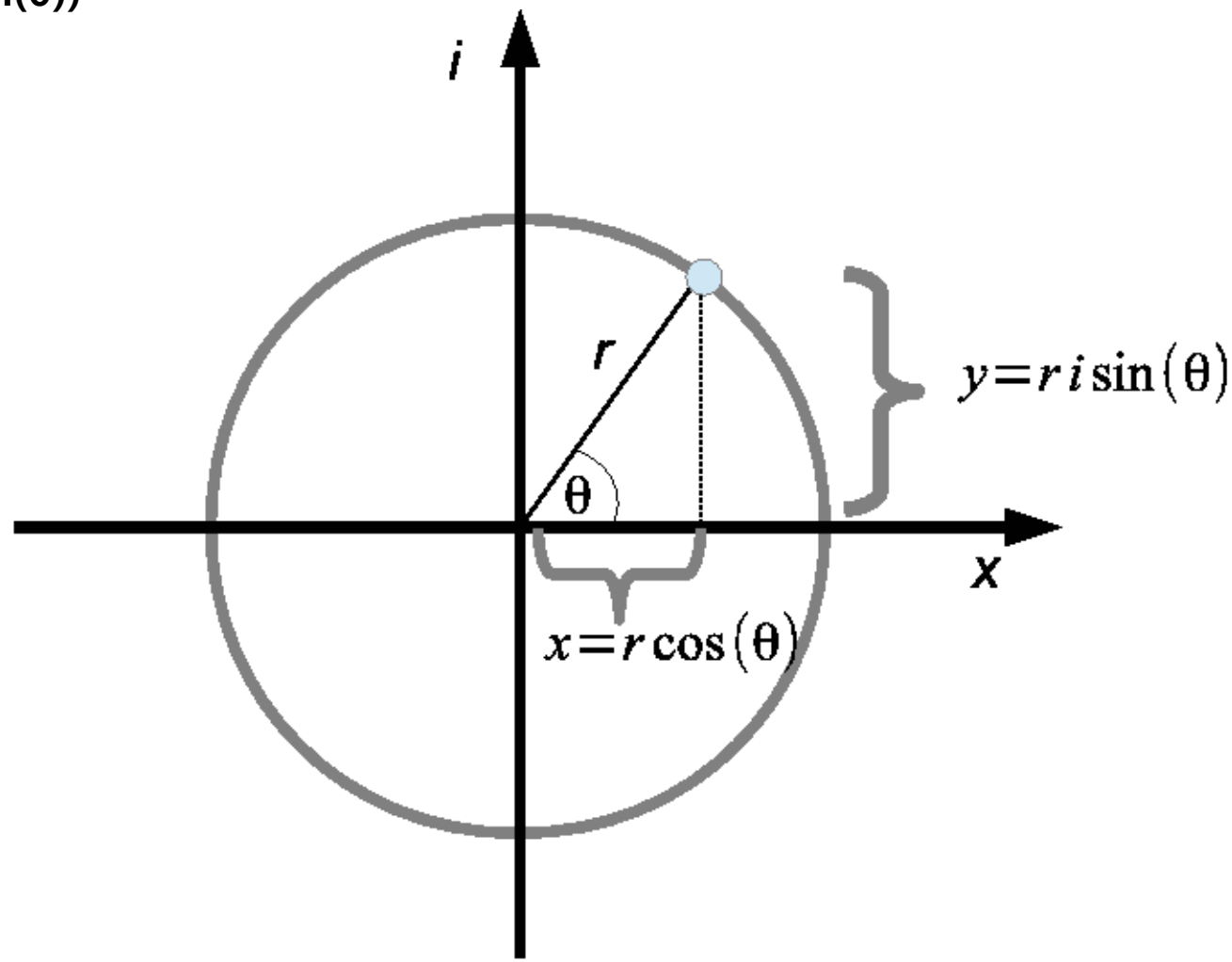
$$\sqrt{(x - a)^2 + (y - b)^2} = r$$

3. Square both sides of the equation $(x - a)^2 + (y - b)^2 = r^2$

https://mathsathome.com/the-equation-of-a-circle/

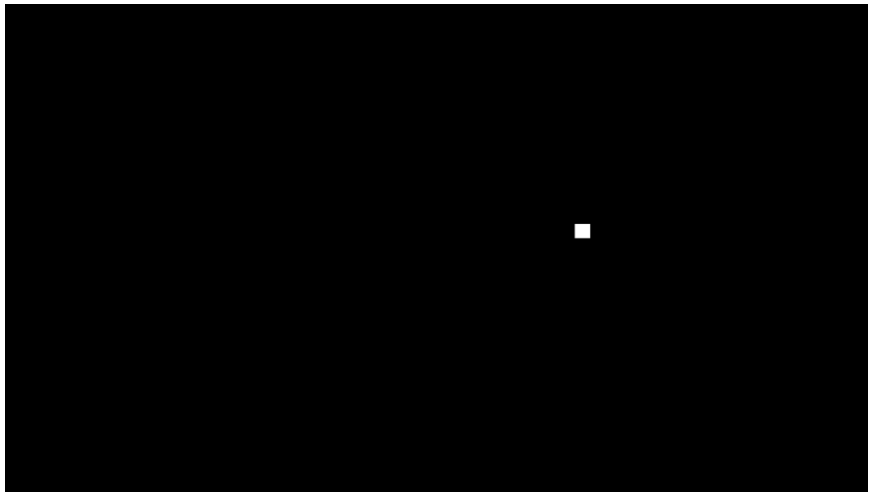
삼각함수를 이용한 원의 표현

- (x, y) = (rcos(θ), rsin(θ))

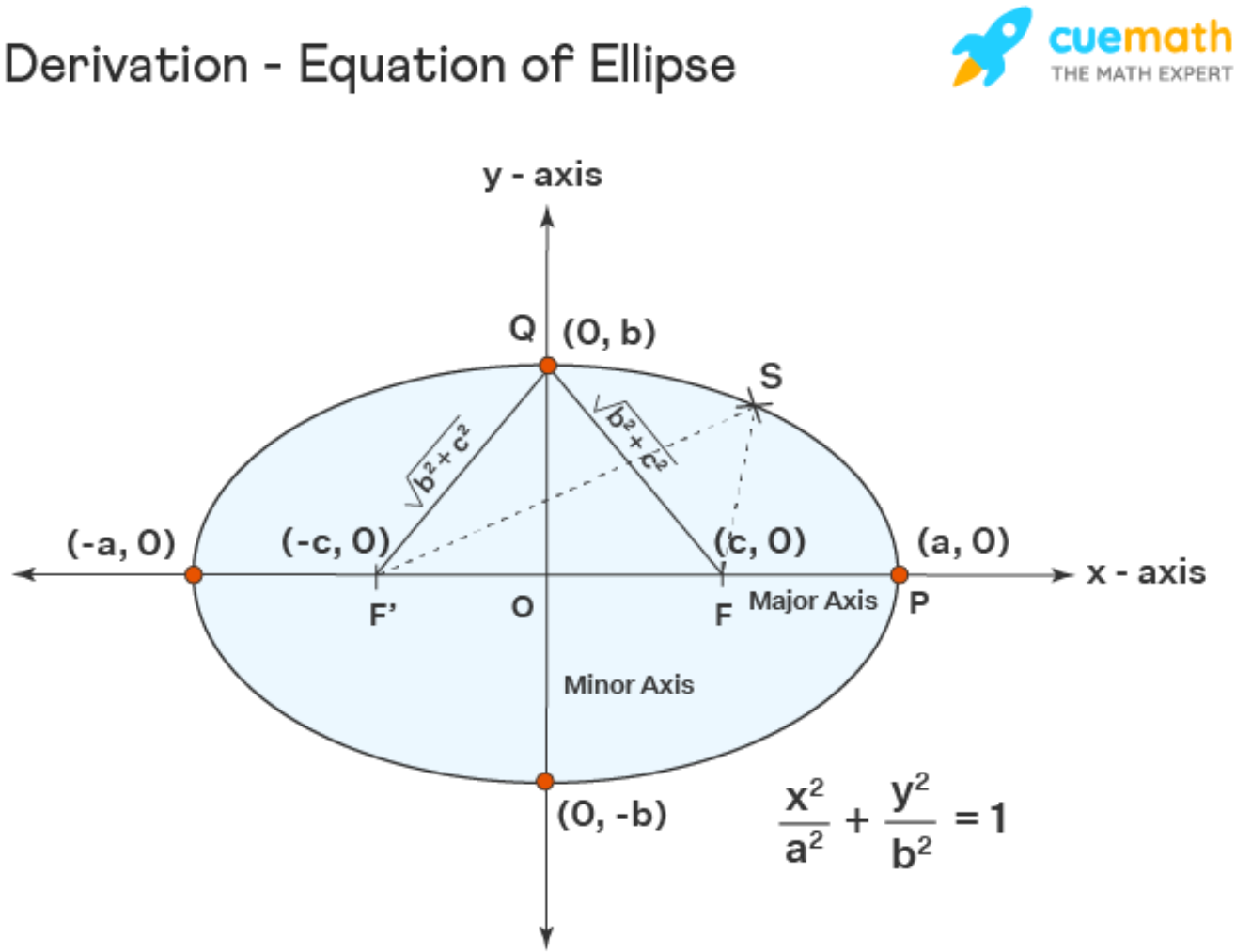


원 운동 (반지름 : RADIUS)

```
using UnityEngine;

public class CircularMovement : MonoBehaviour {
    private float radius = 3f; // 원의 반지름
    private Vector3 center = new Vector3(0, 0, 0); // 원의 중심점
    private float rotationFrequency = 0.5f; // 초당 회전 수
    private float ellipsedTime; // 경과 시간

    private void Start() {
        center = transform.position; // 초기 위치를 중심점으로 설정
    }

    private void Update() {
        ellipsedTime += Time.deltaTime; // 경과 시간 업데이트

        float t = ellipsedTime * rotationFrequency; // 회전 각도 계산
        float x = radius * Mathf.Cos(t * Mathf.PI * 2); // x 좌표 계산
        float y = radius * Mathf.Sin(t * Mathf.PI * 2); // y 좌표 계산
        Vector3 position = new Vector3(x, y, 0); // 새로운 위치 계산
        transform.position = center + position; // 오브젝트 위치 업데이트
    }
}
```



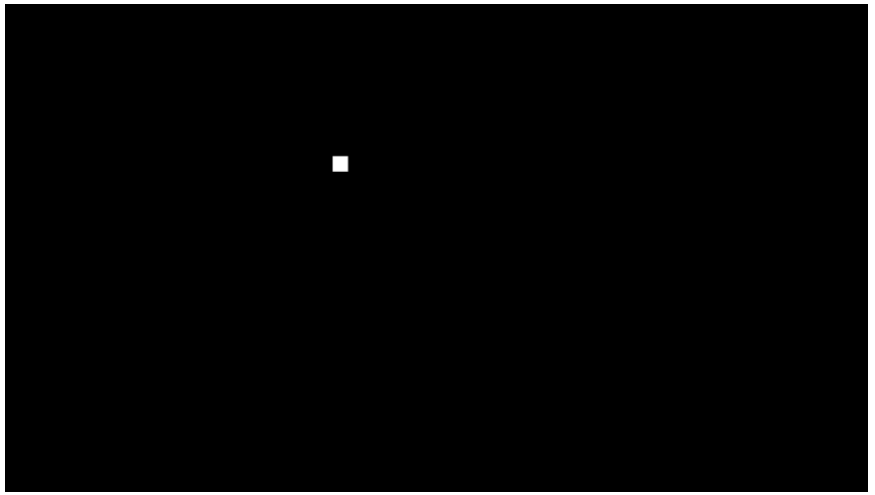
타원의 정의



https://www.cuemath.com/geometry/ellipse/

타원 운동 (장축 : RADISUA / 단축 RADIUSB)

```
using UnityEngine;

public class EllipticalMovement : MonoBehaviour {
    private float radiusA = 4f; // 타원의 장축 반지름
    private float radiusB = 2f; // 타원의 단축 반지름
    private Vector3 center = new Vector3(0, 0, 0); // 원의 중심점
    private float rotationFrequency = 0.5f; // 초당 회전 수
    private float ellipsedTime; // 경과 시간

    private void Start() {
        center = transform.position; // 초기 위치를 중심점으로 설정
    }

    private void Update() {
        ellipsedTime += Time.deltaTime; // 경과 시간 업데이트

        float t = ellipsedTime * rotationFrequency; // 회전 각도 계산
        float x = radiusA * Mathf.Cos(t * Mathf.PI * 2); // x 좌표 계산
        float y = radiusB * Mathf.Sin(t * Mathf.PI * 2); // y 좌표 계산
        Vector3 position = new Vector3(x, y, 0); // 새로운 위치 계산
        transform.position = center + position; // 오브젝트 위치 업데이트
    }
}
```



원 운동 (반지름 RADIUS) + SIN