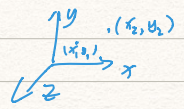
// MeasurementViewController.swift

```
import UIKit
import SceneKit
import ARKit
```



Pythagorean Theorem

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

sqrt((x2 - x1)^2 + (y2 - y1)^2)

```
class viewController: UIViewController, ARSCNViewDelegate{
    var sceneView: ARSCNView!
    var spheres = [SCNNode]()
    override func viewDidLoad(){
        super.viewDidLoad()
        self.sceneView = ARSCNView(frame: self.view.frame)
        self.view.addSubview(self.sceneView)

        sceneView.delegate = self
        sceneView.showsStatistics = true
        let scene = SCNScene()
        addCrossSign()
        registerGestureRecognizers()
        sceneView.scene = scene

    }

    private func registerGestureRecognizers()
        let tapGestureRecognizer = UITapGestureRecognizer(target: self, action: #selector(tapped))
        self.sceneView.addGestureRecognizer(tapGestureRecognizer)
    }

    @objc func tapped(recognizer: UIGestureRecognizer){
        let sceneView = recognizer.view as! ARSCNView
        let touchLocation = self.sceneView.center
        let hitTestResults = sceneView.hitTest(touchLocation, types: .featurePoint)
        if !hitTestResults.isEmpty{
            guard let hitTestResult = hitTestResults.first else {
                return
            }
            let sphere = SCNSphere(radius: 0.005)
            let material = SCNMaterial()
            material.diffuse.contents = UIColor.red
            sphere.firstMaterial = material
            let sphereNode = SCNNode(geometry: sphere)
            sphereNode.position = SCNVector3(hitTestResult.worldTransform.columns.3.x,
                                             hitTestResult.worldTransform.columns.3.y,
                                             hitTestResult.worldTransform.columns.3.z)

            self.sceneView.scene.rootNode.addChildNode(sphereNode)
            self.spheres.append(sphereNode)

            if self.spheres.count == 2{
                let firstPoint = self.spheres.first!
                let secondPoint = self.spheres.last!
```

// must enable plane detection first

// quicker, no anchor

```
let position = SCNVector3Make(secondPoint.position.x - firstPoint.position.x,
                              secondPoint.position.y - firstPoint.position.y,
                              secondPoint.position.z - firstPoint.position.z)
let result = sqrt(position.x * position.x + position.y * position.y + position.z * position.z)
let centerPoint = SCNVector3((firstPoint.position.x + secondPoint.position.x)/2,
                             (firstPoint.position.y + secondPoint.position.y)/2,
                             (firstPoint.position.z + secondPoint.position.z)/2)

display(distance: result, position: centerPoint)

        }
    }
}
private func display(distance: Float, position: SCNVector3){
    let textGeo = SCNText(string: "\(distance)m", extrusionDepth: 1.0)
    textGeo.firstMaterial?.diffuse.contents = UIColor.black
    let textNode = SCNNode(geometry: textGeo)
    textNode.position = position
    textNode.scale = SCNVector3(0.002, 0.002, 0.002)
    self.sceneView.scene.rootNode.addChildNode(textNode)
}

private func addCrossSign(){
    let label = UILabel(frame: CGRect(x:0, y:0, width:100, height:33))
    label.text = "+"
    label.textAlignment = .center
    label.center = self.sceneView.center
    self.sceneView.addSubview(label)
}

override func viewWillAppear(_ animated: Bool){
    super.viewWillAppear(animated)
    let configuration = ARWorldTrackingConfiguration()
    configuration.planeDetection = .horizontal    // planes

    sceneView.session.run(configuration)
}
```

// Google Blocks

// go to the website ⇒ gallary

download blender to convert obj to collada dae

file → import → Wavefront (obj) → model.obj

file → export → collada (dae) → model.dae

make sure the model.dae has Target Membership as the app

// GoogleBlocks.swift

```
import UIKit
import SceneKit
import ARKit
```

```
class viewController: UIViewController, ARSCNViewDelegate{
    var sceneView: ARSCNView!

    override func viewDidLoad(){
        super.viewDidLoad()
        self.sceneView = ARSCNView(frame: self.view.frame)
        self.view.addSubview(self.sceneView)
        sceneView.delegate = self
        sceneView.showsStatistics = true
        let farmScene = SCNScene(named: "farm.dae")!
        let farmNode = farmScene.rootNode.childNode(withName: "farm", recursively: true)

        let scene = SCNScene()
        scene.rootNode.addChildNode(farmNode!)      // adjust the orient   camera -> front

        sceneView.scene = scene
    }
```