1. HitTest

Un hitTest è il risultato che viene restituito quando viene toccato un determinato Trackable. Ogni risultato è costituito da:

- Lunghezza in metri dall'origine del raggio che può essere ricavata dall'invocazione del metodo getDistance().
- Posa (posizione e orientamento) del punto toccato con getHitPose().
- Istanza Trackable che contiene la geometria 3d che è stata toccata con getTrackable()

Questo risultato può essere utilizzato per definire un'ancora che permette di fissare la posizione di contenuti virtuali all'interno dello spazio. L'ancora si adatta agli aggiornamenti dell'ambiente circostante e aggiorna gli oggetti legati ad essa come descritto nel capitolo (Anchor and Trackanle).

Esistono quattro tipi di risultati che si possono ottenere in una sessione ARCore:

- **Profondità**: richiede l'attivazione di depth API nella sessione ARCore ed è usato per posizionare oggetti su superfici arbitrarie (non solo su piani).
- **Aereo**: permette di posizionare un oggetto su superfici piane e utilizza la loro geometria per determinare la profondità e l'orientamento del punto individuato.
- **Punto caratteristico**: permette di disporre oggetti in superfici arbitrarie basandosi su caratteristiche visive attorno al punto sul quale l'utente tocca.
- **Posizionamento istantaneo**: consente di posizionare un oggetto rapidamente in un piano utilizzando la sua geometria completa attorno al punto selezionato.

1.1 Definizione e gestione di un HitTest

E' possibile ricevere un hitTest di tipo diverso nel seguente modo:

```
1
2     //Retrieve hit-test results are sorted by increasing distance from the camera or virtual ray's
3     // origin.
4     val hitResultList =
5         if (usingInstantPlacement) {
6           // When using Instant Placement, the value in APPROXIMATE DISTANCE METERS will determine
7           // how far away the anchor will be placed, relative to the camera's view.
8           frame.hitTestInstantPlacement(tap.x, tap.y, APPROXIMATE_DISTANCE_METERS)
9           // Hit-test results using Instant Placement will only have one result of type
10          // InstantPlacementResult.
11        } else {
12          frame.hitTest(tap)
13        }
14
15    // The first hit result is often the most relevant when responding to user input.
16    val firstHitResult =
17        hitResultList.firstOrNull { hit ->
18         val trackable = hit.trackable!!
19
20         if(trackable is DepthPoint){
21          // Replace with any type of trackable type
22          true
23         }else{
24          false
25         }
26
27        }
28
29    if (firstHitResult != null) {
30        // Do something with this hit result.  For example, create an anchor at this point of interest.
31        val anchor = firstHitResult.createAnchor()
32        //Use this anchor in your AR experience...
33    }
```

Listing 1.1: Filtraggio hitTest in base al tipo

Per definire un hitTest attraverso un raggio **arbitrario** si può usare il metodo **Frame.hitTest(origin3: Array<float>, originOffset: int, direction3: Array<float>, originOffset: int)** dove i quattro parametri specificano:

- *origin3*: array che contiene le 3 coordinate del punto di partenza del raggio.
- *originOffset*: offset sommato alle coordinate dell'array di partenza.
- *director3*: array che contiene le 3 coordinate del punto di arrivo del raggio.
- *directorOffset*: offset sommato alle coordinate dell'array di arrivo.

Per creare un anchor sul risultato del tocco viene usato **hitResult.createAnchor()** che restituirà un anchor disposto sul Trackable sottostantesu cui è venuto il tocco.
Nel caso della nostra applicazione il risultato restituito da hitTest nella modalità *Plane Detection* è di tipo Aereo; il rilevamento di un piano consente di disporre un animale in un punto preciso. Questo evento è stato gestito dal metodo *setOnTapArPlaneListener* riportato nell'esempio di codice **??** a pagina ??.