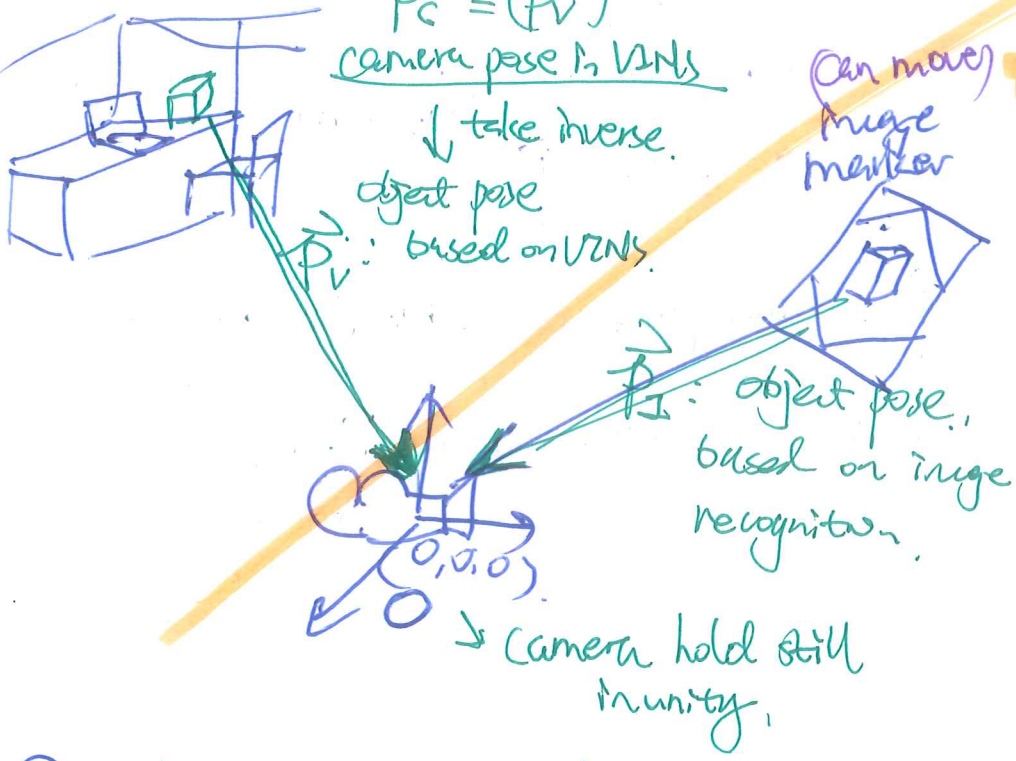
① think camera as reference frame (world)

$\vec{P}_C = (\vec{P}_V)^{-1}$

camera pose in VINS

↓ take inverse.

object pose based on VINS



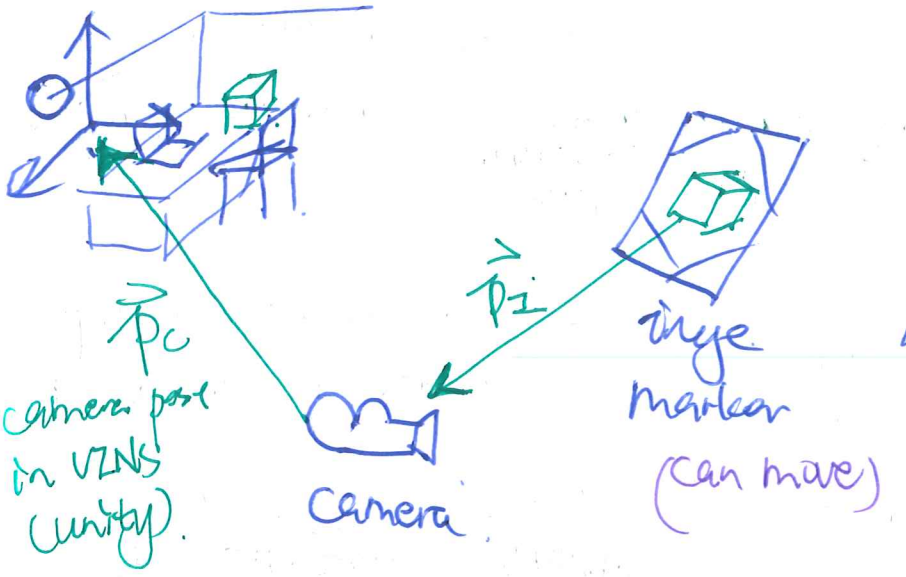
↳ camera hold still in unity.

two system

seperated: independent of each other.

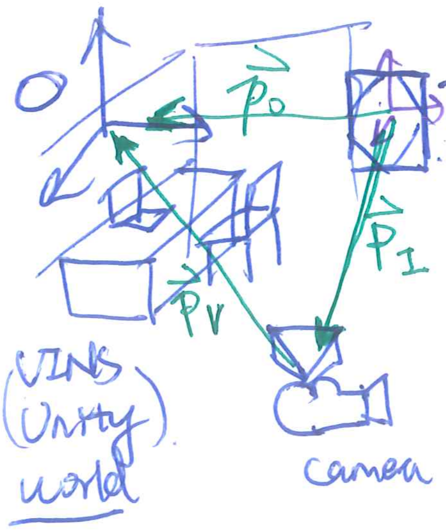
② think VINS world as reference frame (world)

(assuming VINS reference is the same as unity)



seems to be ~~one~~ one system., by ~~it~~ still two seperated system

③ combine image pose to correct VINS



→ marker image: hold still in world (VINS/Unity), has pose $\vec{p}_0$

$\vec{p}_I$: image pose find with image marker

$\vec{p}_v$: camera pose find by VINS.

(*)

① with $\vec{p}_v$ → get camera pose obviously $\boxed{\vec{p}_v}$

② with $\vec{p}_I$ and $\vec{p}_0$ → camera pose should be

$$\boxed{\vec{p}_0 (\vec{p}_I)^{-1}}$$

If no error: $\vec{p}_c = \vec{p}_v = \vec{p}_0 (\vec{p}_I)^{-1}$ ← can also seen from the upper image

($\vec{p}_c$: camera real pose; $\vec{p}_v$: VINS pose; $\vec{p}_0$: image set pose; $\vec{p}_I$: Image localization pose)

(*) now, $\vec{p}_v$ has error.

$$\Rightarrow \vec{p}_c = \vec{p}_v + \vec{err} = \vec{p}_0 (\vec{p}_I)^{-1}$$

← we can use the left function to correct $\vec{p}_v$

(1) 如果用之前用观点，用图像定位 则直接赋值 $\vec{p}_{v_0} = \vec{p}_0 (\vec{p}_I)^{-1}$ （时刻为t0）

然后继续VINS追踪，则这时 ~~VINS~~ 时刻为t 所输出应该为：

$$\tilde{p}_{ct} = \vec{p}_{v0} (\vec{p}_{v0})^{-1} \vec{p}_{vt}$$

相机位姿估计 → $\tilde{p}_{ct}$

$\vec{p}_{v0}$：图像识别时重新初始化.

$(\vec{p}_{v0})^{-1} \vec{p}_{vt}$：t0到t, VINS测量到的相对位移.

(2) 但是实际情况不是这样的

因为 $\vec{r}_1$ 和 $\vec{r}_2$ 都有误差.
(VINS) (Image)

实际应该是 $\vec{r}_C = \vec{r}_1 + \vec{e}_1 = \vec{r}_0 (\vec{r}_2 + \vec{e}_2)^{-1}$
Camera

→ 直接用图像和地心是不理想的

应该是 $\tilde{\vec{r}}_C = f(\vec{r}_1, \vec{r}_2)$
抽象的估计

比如取平均: $\tilde{\vec{r}}_C = \left[\vec{r}_1 + \vec{r}_0 (\vec{r}_2)^{-1}\right] \frac{1}{2}$.

实际使用 Kalman Filter 卡尔曼滤波来处理是比较理想的.