深蓝学院
shenlanxueyuan.com

视觉SLAM基础与VIO进阶
第十章作业分享

主讲人 方块块块

VIO文献阅读

- 视觉与IMU进行融合之后有何优势？

这一问题高博在视频中讲解的很清楚，在视频中的对比如下图所示：

IMU 与视觉定位方案优势与劣势对比：

方案	IMU	视觉
优势	快速响应 不受成像质量影响 角速度普遍比较准确 可估计绝对尺度	不产生漂移 直接测量旋转与平移
劣势	存在零偏 低精度 IMU 积分位姿发散 高精度价格昂贵	受图像遮挡、运动物体干扰 单目视觉无法测量尺度 单目纯旋转运动无法估计 快速运动时易丢失

VIO文献阅读

深蓝学院
shenlanxueyuan.com

● 视觉与IMU进行融合之后有何优势？

而在论文中也有类似叙述该结论的部分，如下：

(IMU) sensor. These sensors are made available nowadays with high accuracy, miniaturised size, and low cost because of the fast-developing manufacturing of chips and micro-electromechanical systems (MEMS) devices. And they are complimentary with one another in a way which would be able to compensate for the errors made by each of them via the redundant information they provided. Furthermore, the

由于芯片和微电子机械系统(MEMS)设备的快速发展，IMC现在能以高精度、小型化和低成本提供。他们(视觉与IMU)是相互互补的，能够通过他们提供的冗余信息来弥补各自的不足。

VIO文献阅读

- 视觉与IMU进行融合之后有何优势？

综上所述，视觉和 IMU 在很大程度上是互补的，他们能够互相弥补各自的短板，IMU 响应速度快，一般都大于 100Hz，且不受太多其他环境因素的影响，但会存在零偏。而视觉 不怎么会产生漂移，但缺点比如受图像质量、环境影响，而且单目时无法测量绝对尺度，快 速运动时也很容易出问题。它们二者提供的冗余信息可以弥补各自的不足。

VIO文献阅读

●有哪些常见的视觉+IMU 融合方案？有没有工业界应用的例子？

在“Visual and Visual-Inertial SLAM: State of the Art, Classification, and Experimental Benchmarking”一文中对今年来的vSLAM和viSLAM做出一些比较分类。分类上可以分为紧耦合和松耦合的方法，表格还列出了他们视觉部分采用的相机种类（单目？双目？），以及基于滤波的角度还是基于优化的角度。

（实际上我在做作业的时候貌似参考的是另一篇只有viSLAM总结而没有vSLAM的文献，但分享的时候没有找到。。也说明做好标注来源的重要性，如果同学知道还请分享。。。）

Algorithm map gestion	Hardware requirements				Approach			Input treatment		Localis./Mapping			Memory loop	
	Monoc.	Stereo	Depth	IMU	Filter	Optim.		Direct	Indir.	2D-2D	3D-2D	IMU	closure	
MonoSLAM [21]	X				X		Sparse		X		X			
Monocular FastSLAM [22]	X				X		Sparse		X		X			
PTAM [27]	X					X	Sparse		X		X			
PTAM with edgelets [65]	X					X	Sparse		X		X			
PTAM with DWO [79]	X					X	Sparse		X		X			X
Stereo PTAM [78]		X				X	Sparse		X	X	X			X
CD-SLAM [80]	X					X	Sparse		X		X			X
ORB-SLAM [37]	X					X	Sparse		X	X	X		X	X
ORB-SLAM2 [76]	X	(X)	(X)			X	Sparse		X	X	X		X	X
Edge-SLAM [81]	X					X	Sparse		X	X	X			X
DTAM [34]	X					X	Dense	X		(X)	X			
MobileFusion [66]	X			(X)		X	Dense	X			X	(X)	X	
Semidense visual odom. [5]	X					X	Semidense	X		X			X	
LSD-SLAM [35]	*X*					*X*	*Semidense*	*X*		*X*				*X*
Semidirect VO (SVO) [67]	X					X	Sparse	X	X	X	X		X	
Direct sparse odom. (DSO) [33]	*X*					*X*	*Sparse*	*X*		*X*			*X*	
KinectFusion [68]			X			X	Dense	X			X			
Kintinuous [82]	(X)		X			X	Dense	X			X			X
DVO SLAM [69]	X		X			X	Dense	X		X				X
ElasticFusion [70]	X		X			X	Dense	X			X		X	X
MSCKF [25]	X			X	X		None		X		X	X	X	
MSCKF 2.0 [45]	X			X	X		None		X		X	X	X	
ROVIO [26]	X	(X)		X	X		None	X	X	X		X	X	
OKVIS [73]	(X)	X		X		X	Sparse		X	X	X	X	X	
S-MSCKF [17]		X		X	X		None		X		X	X	X	
Vins-Mono [74]	X			X	X		Sparse		X		X	X	X	X
Kimera [60]	(X)	X		X	X	X	Dense		X	X	X	X	X	X
SOFT-SLAM [72]		X		(X)		X	Dense		X	X	X	(X)	X	X
STCM-SLAM [77]		X		X		X	Sparse		X		X	X	X	X
VIORB [75]	X			X		X	Sparse		X	X	X		X	X

VIO文献阅读

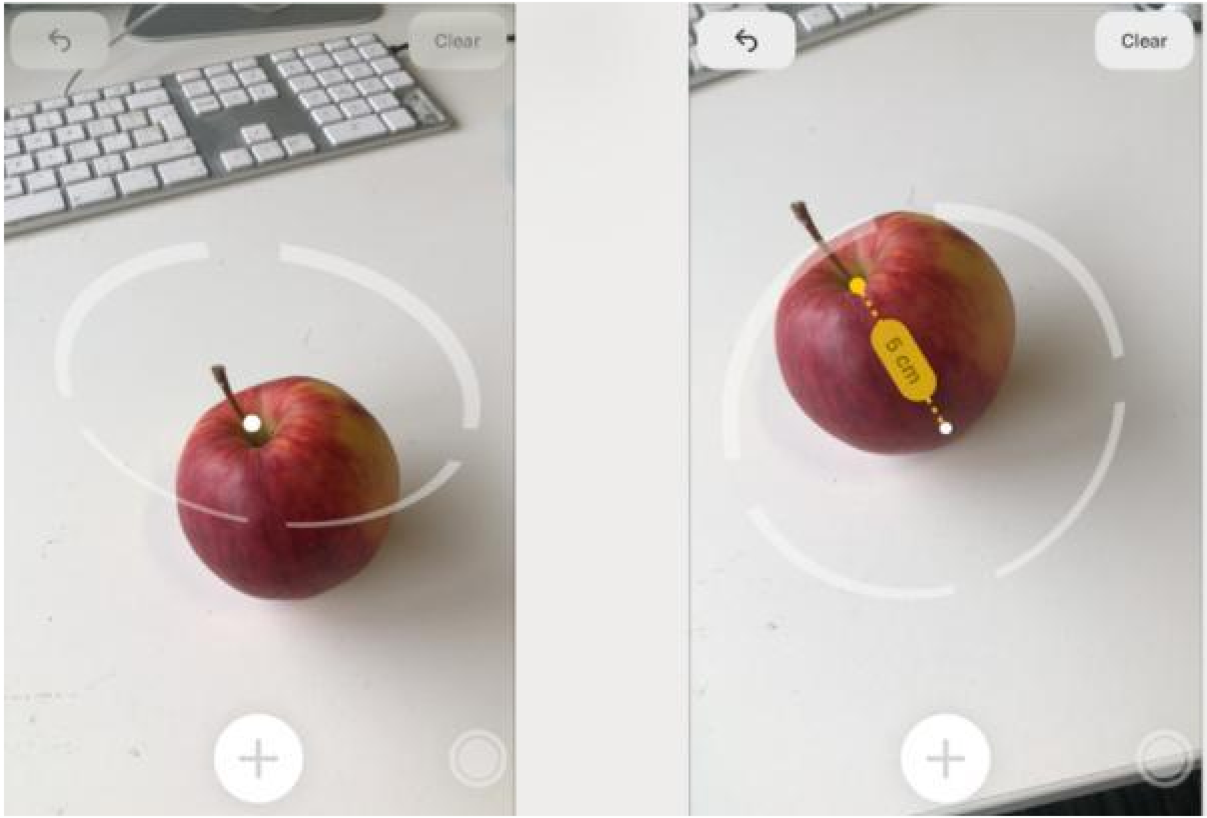
深蓝学院 shenlanxueyuan.com

●有哪些常见的视觉+IMU 融合方案？有没有工业界应用的例子？

关于工业界应用的例子，广范来讲，应用在AR领域、无人机领域、自动驾驶和室内机器人等方向，包括大疆的无人机，谷歌的Tango和ARcore，百度的DuMix AR，苹果的ARKit等。

后面三个都是各家公司推出的AR开发平台，这里有一个例子，苹果在IOS12推出了一款使用ARKit开发的APP“Measure”，能够实现基于AR的测量工作。

VIO文献阅读



VIO文献阅读

- 学术界的VIO研究有哪些新进展？有没有将学习方法用到其中的例子？

我参考了知乎大佬在19年的一份总结：https://zhuanlan.zhihu.com/p/68627439

还是相对比较全面的，除了文章题目还给出了一些大致的内容总结和开源地址，和助教分享的几篇也有不小的重合度。

四元数和李代数更新

深蓝学院 shenlanxueyuan.com

2. 四元数和李代数更新

课件提到了可以使用四元数或旋转矩阵存储旋转变量。当我们用计算出来的 ω 对某旋转更新时，有两种不同方式：

$$\begin{aligned} &\mathbf{R} \leftarrow \mathbf{R}\exp\left(\omega^{\wedge}\right) \\ \text{或}\quad &\mathbf{q} \leftarrow \mathbf{q} \otimes \left[1, \tfrac{1}{2}\omega\right]^{\mathrm{T}} \end{aligned} \tag{20}$$

请编程验证对于小量 $\omega = [0.01, 0.02, 0.03]^{\mathrm{T}}$，两种方法得到的结果非常接近，实践当中可视为等同。因此，在后文提到旋转时，我们并不刻意区分旋转本身是 $\mathbf{q}$ 还是 $\mathbf{R}$，也不区分其更新方式为上式的哪一种。

```cmake
cmake_minimum_required(VERSION 3.10)
project(verify)

# Eigen
include_directories("/usr/local/include/eigen3")

# Sophus
find_package(Sophus REQUIRED)
include_directories(${Sophus_INCLUDE_DIRS})

add_executable(verify verify.cpp)
target_link_libraries(verify Sophus::Sophus fmt)
```

```cpp
#include <iostream>
#include <eigen3/Eigen/Core>
#include <eigen3/Eigen/Geometry>
#include <sophus/so3.hpp>
#include <sophus/se3.hpp>

using namespace std;
using namespace Eigen;
using namespace Sophus;

int main() {
    // 定义旋转矩阵与四元数
    AngleAxisd rotation_vector(M_PI/4, Vector3d(0,0,1));
    Matrix3d R = rotation_vector.toRotationMatrix();

    Quaterniond q = Quaterniond (R);
    cout << "初始旋转矩阵: " << endl << R << endl;
    cout << "初始四元数: " << endl << q.coeffs() << endl;

    // w ---- 微小角速度
    Vector3d w(0.01,0.02,0.03);

    // 第一种方式
    SO3d SO3_R(R);
    SO3d SO3_new = SO3_R * SO3d::exp(w);
    cout<<"第一种方法的结果: "<< endl << SO3_new.matrix() <<endl;

    // 第二种方式
    Eigen::Quaternion q_w(1,w(0)/2,w(1)/2,w(2)/2);
    Eigen::Quaternion q_new = q*q_w;
    q_new = q_new.normalized();
    cout << "第二种方法的结果: " << endl << q_new.toRotationMatrix() << endl;

    return 0;
}
```

四元数和李代数更新

2. 四元数和李代数更新

课件提到了可以使用四元数或旋转矩阵存储旋转变量。当我们用计算出来的 ω 对某旋转更新时，有两种不同方式：

$$\begin{aligned} &\mathbf{R} \leftarrow \mathbf{R}\exp\left(\omega^{\wedge}\right) \\ \text{或} \quad &\mathbf{q} \leftarrow \mathbf{q} \otimes \left[1, \tfrac{1}{2}\omega\right]^{\mathrm{T}} \end{aligned} \tag{20}$$

请编程验证对于小量 $\omega = [0.01, 0.02, 0.03]^{\mathrm{T}}$，两种方法得到的结果非常接近，实践当中可视为等同。因此，在后文提到旋转时，我们并不刻意区分旋转本身是 q 还是 R，也不区分其更新方式为上式的哪一种。

```
初始旋转矩阵:
 0.707107 -0.707107          0
 0.707107  0.707107          0
        0         0          1
初始四元数:
       0
       0
0.382683
 0.92388
第一种方法的结果:
  0.685368   -0.727891   0.0211022
  0.727926    0.685616 0.00738758
-0.0198454   0.0102976     0.99975
第二种方法的结果:
  0.685371   -0.727888   0.0210998
  0.727924    0.685618 0.00738668
-0.0198431   0.0102964     0.99975
*** Finished ***
```

右乘模型

3. 使用右乘 so(3)，推导以下导数： $\dfrac{d(R^{-1}p)}{dR}$ ， $\dfrac{d\ln(R_1R_2^{-1})^\vee}{dR_2}$

解：已知旋转点的右扰动雅可比：

$$\frac{\partial(Rp)}{\partial\varphi} = \lim_{\varphi\to 0}\frac{R\exp(\varphi^\wedge)p - Rp}{\varphi}$$

$$= \lim_{\varphi\to 0}\frac{R(I+\varphi^\wedge)p - Rp}{\varphi}$$

$$= \lim_{\varphi\to 0}\frac{R\varphi^\wedge p}{\varphi} = \lim_{\varphi\to 0}\frac{-Rp^\wedge\varphi}{\varphi}$$

$$= -Rp^\wedge$$

则

$$\frac{d(R^{-1}p)}{dR} = \frac{\partial(R^{-1}p)}{\partial\varphi}$$

$$= \lim_{\varphi\to 0}\frac{[R\exp(\varphi^\wedge)]^{-1}p - R^{-1}p}{\varphi}$$

$$= \lim_{\varphi\to 0}\frac{\exp(\varphi^\wedge)^{-1}R^{-1}p - R^{-1}p}{\varphi}$$

$$= \lim_{\varphi\to 0}\frac{[I-\varphi^\wedge - I]R^{-1}p}{\varphi}$$

$$= \lim_{\varphi\to 0}\frac{(R^{-1}p)^\wedge\varphi}{\varphi} = (R^{-1}p)^\wedge$$

$$\frac{d\ln(R_1R_2^{-1})^\vee}{dR_2} = \lim_{\phi\to 0}\frac{\ln[R_1(R_2\exp(\phi^\wedge))^{-1}]^\vee - \ln(R_1R_2^{-1})^\vee}{\phi}$$

$$= \lim_{\phi\to 0}\frac{\ln[R_1\exp(\phi^\wedge)^{-1}R_2^{-1}]^\vee - \ln(R_1R_2^{-1})^\vee}{\phi}$$

[要凑 $\ln[R_1R_2^{-1}\cdots]^\vee$]

$$= \lim_{\phi\to 0}\frac{\ln[R_1R_2^{-1}R_2\exp(\phi^\wedge)^{-1}R_2^{-1}]^\vee - \ln(R_1R_2^{-1})^\vee}{\phi}$$

$$= \lim_{\phi\to 0}\frac{\ln[R_1R_2^{-1}R_2\exp(-\phi^\wedge)R_2^T]^\vee - \ln(R_1R_2^{-1})^\vee}{\phi}$$

[对旋转矩阵 $a' = R^{-1}a = R^Ta$]

又∵ SO(3)伴随性质，即 $R\exp(p^\wedge)R^T = \exp((Rp)^\wedge)$

$$\therefore \frac{d\ln(R_1R_2^{-1})^\vee}{dR_2} = \lim_{\phi\to 0}\frac{\ln[R_1R_2^{-1}\exp((-R_2\phi)^\wedge)]^\vee - \ln(R_1R_2^{-1})^\vee}{\phi}$$

$$= \lim_{\phi\to 0}\frac{J_r^{-1}(\ln(R_1R_2^{-1})^\vee)(-R_2\phi)}{\phi}$$

$$= -J_r^{-1}(\ln(R_1R_2^{-1})^\vee)R_2$$

深蓝学院
shenlanxueyuan.com

感谢各位聆听!

Thanks for Listening