

# 项目报告

**Delivery of a project in**

**Image**

**Analysis**

**anD**

**Computer**

**Vision**



Lorenzo GUERRIERI Hugo MARINE

**Lucas FOUREST**

学年： 2022-2023

# Project context and global methodology

对于这个项目，我们有一些定制的乒乓球视频，每个视频都涉及不同的游戏系列。 我们选择一个特定的，以便执行一些关键的计算机视觉任务（如跟踪和重建），我们拍摄的每个游戏序列，仅限于一个视角。这里我们选择一段两球员比赛的视频，从后面拍摄的轻微侧面视角， 以下所示：



Figure 1: Point of view

起初，我们要开发一种方法来追踪到这两个玩家手上的球拍和球。然后，作为逻辑的延续，我们将尝试恢复球的真实位置，考虑到我们只有一个视角，不可能使用任何三角测量triangulation。而这部分的挑战是寻找三角测量的另一种替代方法，来克服这一局限性。

对于这里的项目，我们将主要使用 *Python* 和 OpenCV库，及 *numpy* 。

# 1 | 移动物体追踪

## 球拍追踪

区分物体的第一个明显策略是通过颜色来区分它们。在这里，球拍是显而易见的，其颜色也清晰可见：此种策略在以下用例中是可靠的：

然而，值得注意的是，BGR 表示对于颜色过滤来说并不是最佳的：正如 SIMSE Gokhan 所说，我们应该进入 HSV 领域在 **[2]** .

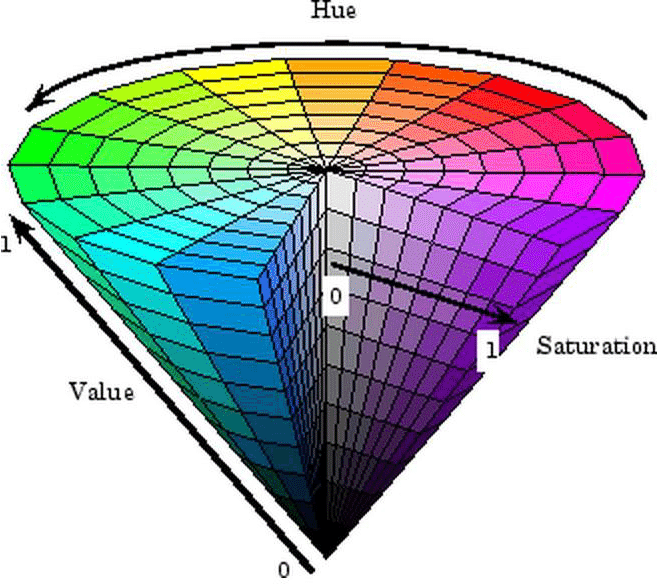


Figure 1.1: HSV space

然而，很难一眼看出三维BGR向量颜色，对应的 HSV 表示，因此我们将依赖于纯粹的计量方法：在鼠标指向位置，打印出某个像素的HSV值。

这样就可在图像上估量出 HSV 上下限过滤蒙版。该颜色过滤蒙版应用于图像，并将球拍从其余元素中分离出来，然后用一个简单的轮廓检测算法，就可以追踪球拍的边界 。



图 1.2: 球拍追踪

## 球的追踪

我们必须采用不同的方法来追踪求。确实，我们希望能继续走同样的路， 但球很小，移动速度很快，且在视频中表面颜色会因受到曝光和亮度的影响而变化。这些条件将不再满足基于颜色的解决方案。

因此我们使用一种背景扣除法来快速识别移动对象。它参考最近 N 帧视频，计算出视频的“平均帧”，其中 N 是一个可调参数，如代码中所述。 然后，它使用高斯混合模型（GMM），计算出每个坐标的最后 N 帧的“平均像素”。 然后平均帧被“删除”。 最后，根据与平均帧相减后的绝对值，将“前景”像素（与移动对象相关）与“背景”像素区分开来，并将其与某个阈值 T（又是一个可调参数）进行比较。

Here an overall view of what it does：



图 1.3: 前景检测

做完这些， 留给我们一个仍是相当嘈杂的二进制（黑白）图像， 因此我们使用morphological

形态学操作来执行“侵蚀”和“膨胀”，以便消除少量不相关像素。 更切确的说， 我们先应用一个closing 操作（先扩张dilatation 后侵蚀erosion） ，然后是一个open操作（先侵蚀后扩张）。

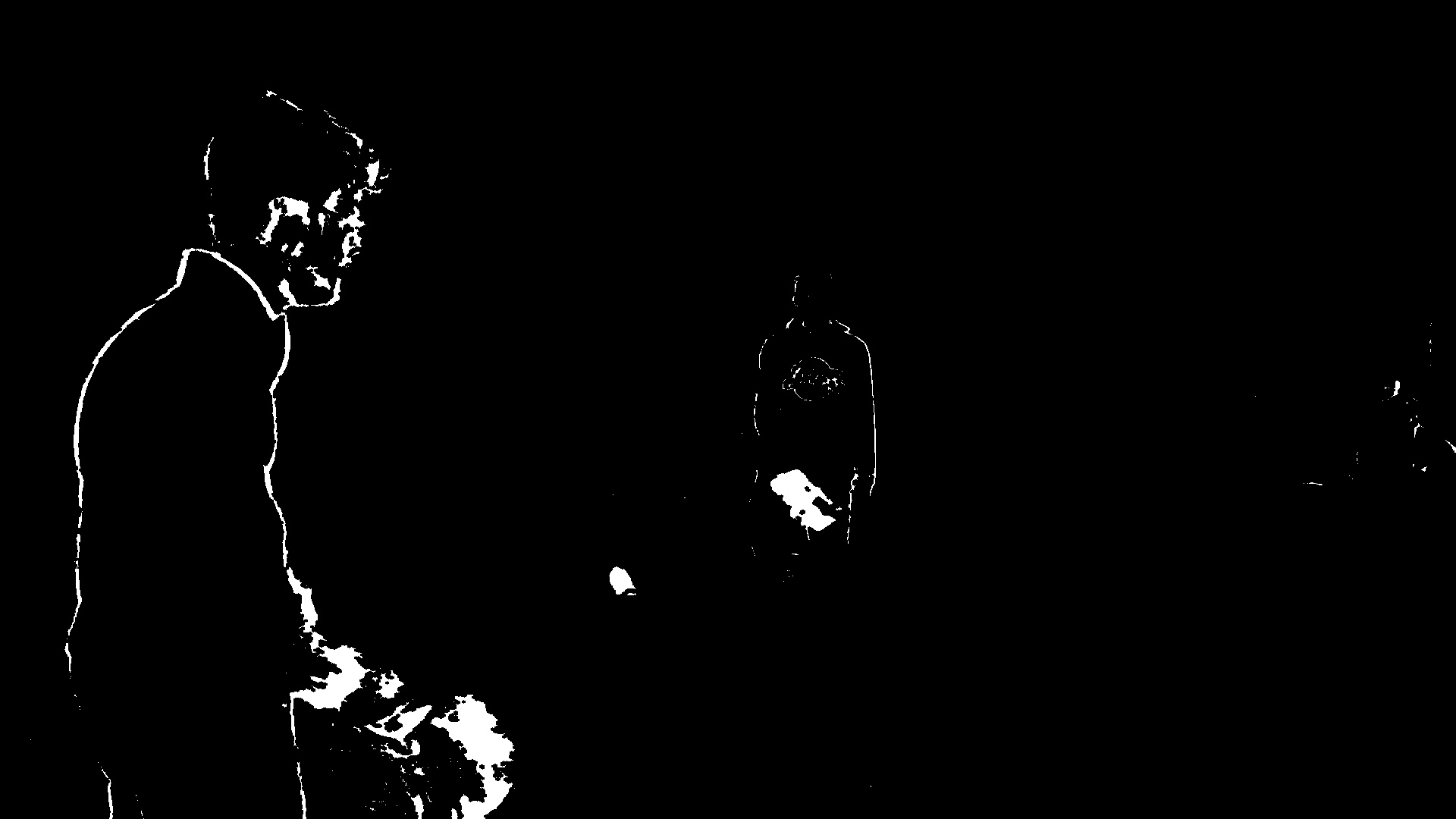


图 1.4: After closing

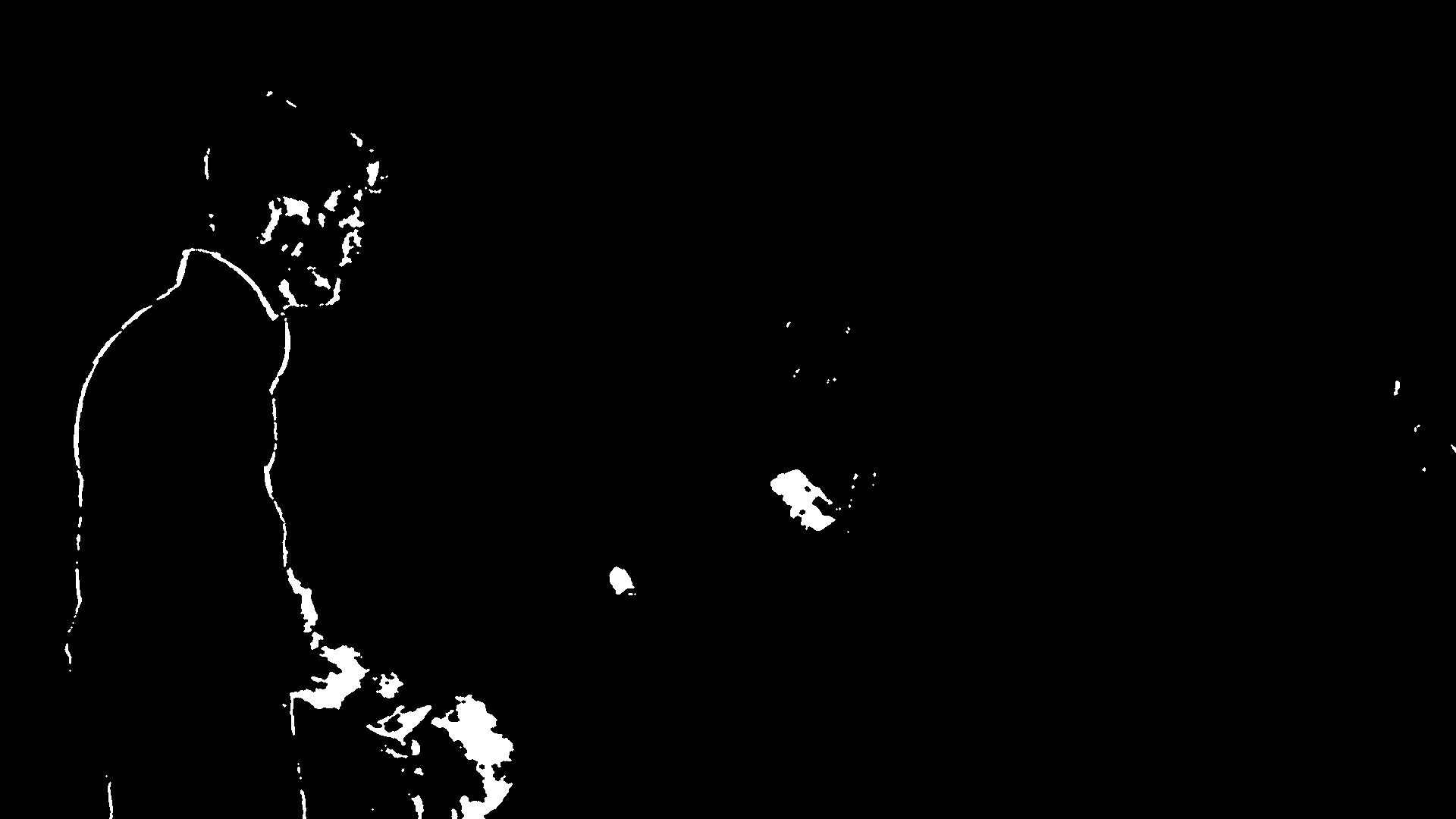


图 1.5： After closing-opening

如 **[1] 中**所示, 扩张和侵蚀 是形态学操作，两个都 应用在*滑动*"sliding" *于*二进制图像X（通常是黑白）上*的*任意形状B上*.* 在数学上：

* 对*X的B*侵蚀记为*ϵ B* ( *X* )，被定义为：对于点*x* ∈ *X*的任何点，B为*x的*中心 （记为*Bx* ）应完全包含在*X中*，意味着对于*Bx* 中的每一个“ 像素”都位于图像X中：

*ϵB*(*X*) = *X ⊖ B* = (*x ∈ X|Bx ⊂ X*)

* *对X*的B膨胀记为*δ B* ( *X )，*与腐蚀操作相反：定义为 点*x* ∈ *X*的任何ensemble点使得*B x 与X*有非空交集，意味着在*Bx* 中 至少有 ” 1个 像素"位于图像 *X中*：

*δB*(*X*) = *X ⊕ B* = (*x ∈ X|Bx ∩ X* = *∅*)

close操作有助于将附近的区域分组为单个对象。因此，有⼀些彼此靠近的较⼤区域应该在我们做其他事情之前连接起来。 这可能是先执行close的原因。 然后进行open操作，这样就可以去除孤立的噪声区域。通常close 结构元素的大小较大，因为我们要确保获得附近的像素，而open结构元素尺寸较小，这样我们就不会错误地删除任何 更大地区。下图直观地总结了⼀切：

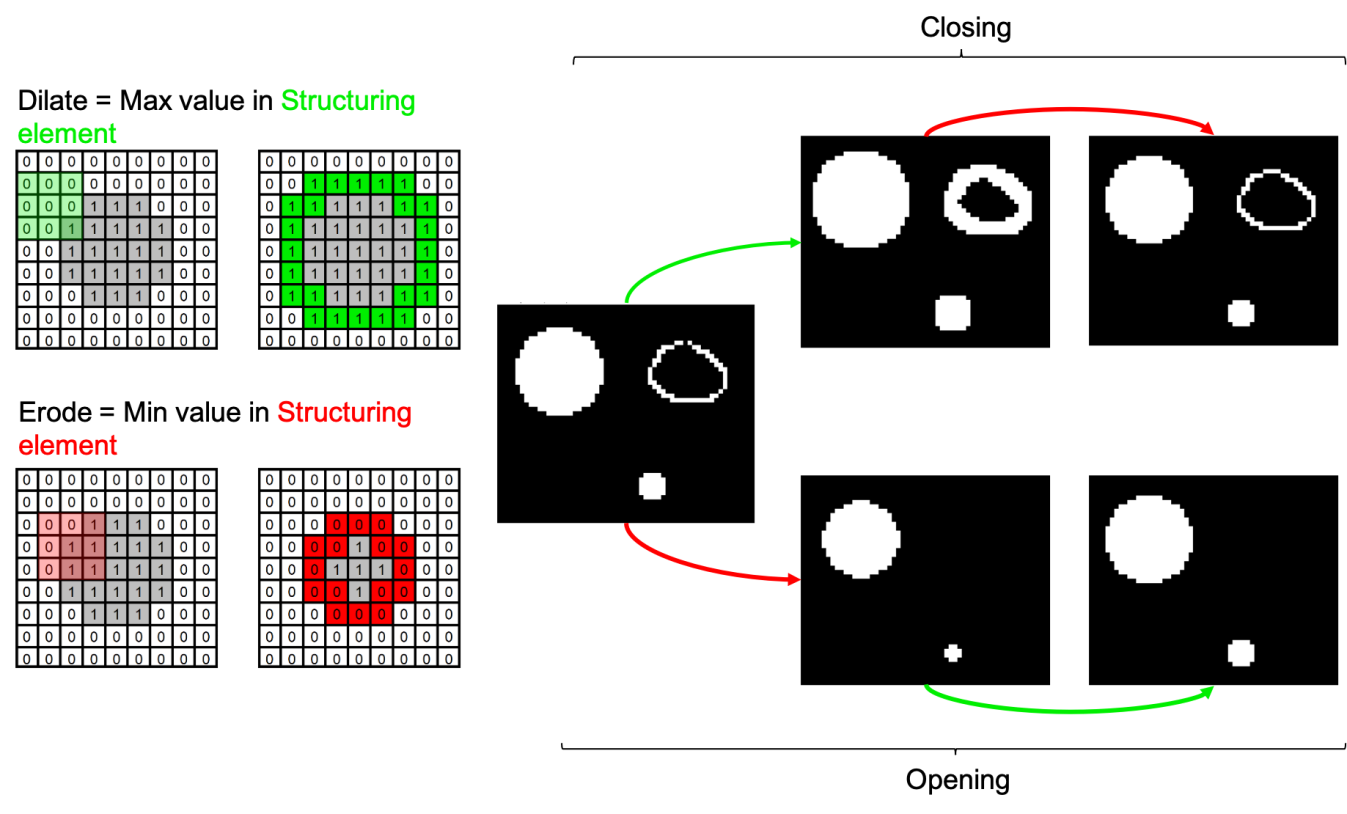


图 1.6: 形态学操作

整个close-open过程应该有效地仅保留最相关的像素斑点（连接白色像素的数量）从而隔离球，作为⿊⽩图像上出现的最后斑点之⼀。

最后，我们在所获得的图像上，使⽤openCV具有⾃定义可调参数的斑点blob检测器来捕获（⼏乎仅）球像素， 并在原始图像上绘制相应的轮廓。



图 1.7: 结果

## 结果讨论

如结果视频中所述，这里的球拍跟踪效果非常好，如结果视频中所述，这主要是因为球拍的颜色非常具体，而且在摄像机上始终清晰可⻅。

然而，尽管在去噪方面付出了很多努力，球跟踪仍然不完善，有时⽆法检测到球，或者受到检测到的不相关斑点的⼲扰。这 是整体视频条件较差与小且快速移动的球相结合的结果，有时在某些帧上几乎“看不见”。我们应该意识到项⽬其余部分的这⼀弱点，因为下面将依赖于球跟踪的质量,来作为输⼊序列。

（a） 没检测到 （b） 检测到噪声

图 1.8： 弱点

除了这些系列点的[x,y]外，我们还会存储对应序列的 时间戳*t* ，以备之后使用。最终我们想要得到所有时刻的*球的实时*[ *X, Y, Z ] 位置*。

# 2 | 球轨迹 3D 重建

## 单视图局限性：替代方法

通常，为了恢复视频移动对象的真实位置，我们⾄少需要该对象连续位置的两个不同视点，并使⽤这些多个点序列来执⾏三⻆测量。在这⾥，每个视频都会拍摄不同的游戏序列（我们仅有⼀台摄像机），这使得三⻆测量变得不可能。

尽管如此，根据文献**[4]** ，可使用单个视角和对象运动属性的附加信息，来重建球的3D击打轨迹。 事实上，正如我们稍后将更详细地描述的那样：每次击球都涉及通用运动方程，我们可利用exploit它来重写一些关联，并且仍可以（大约）用一个视角，来恢复[ *X, Y, Z* ]现实世界位置，只要找到一组完全确定（真实）轨迹的物理参数（称为起始位置、速度 和 加速度）。

该方法仅依赖于投影矩阵*P*系数，一系列2D 点和对应的时间戳。因此我们首先需要找到 和计算出 P， 然后我们就拥有了应⽤上述⽅法所需的⼀切。

## 清理输入顺序

如前所述， 2D 点[ *x, y* ]的输入序列，应仅表示**单次**shot对应的时间点（例如，连续两次表面球接触之间的时间间隔）。因此我们将只保留视频精⼼挑选的部分：在此期间，球以单个抛物线运动。在这部分视频中，跟踪算法返回的点序列才是我们的输⼊序列。对精⼼挑选的不同视频，可重复该过程，来重建shot轨迹。

但我们坚持这样一个事实：所选择的点应该属于**单个**shot序列，因此，考虑到我们的局限性，迫使我们将重建限制到一个单次shot阶段（在反弹之间），我们必须定义和界定要使用视频中的一小部分作为有用部分（将将其点的序列自动 '清理' ，正如现在所见的）。

现在让我们将*S* = ([ *x, y* ] *t* 1 *,* [ *x, y* ] *t* 2 *...* [ *x, y* ] *t n* )表示为我们所谓的输入点序列。由于跟踪缺陷，这可能包含一些噪声点，及捕获良好但不属于单次shot轨迹所考虑的点（如在我们试图重建的shot的两个反弹点之外的点）。这些“异常点”可能会导致重建出现错误，应从 *S中* 删除*。* 但是我们必须找到一种方法来区分和检测这些异常点。

我们将使用以下基于 RANSAC 的方法：其想法是“内点”点 应该都或多或少地，位于二维抛物线曲线上，因为原始的真实运动是抛物线。三点是定义抛物线的最⼩集合，我们将迭代：

* 选择一个子集 ( p1, p2, p3） ∈ S，
* 使⽤最⼩⼆乘法通过该最⼩集拟合抛物线 P，
* 收集S中位于P上的所有点 *p* = [ *x, y* ]]，且误差可接受（最多 范围*ϵ* : ||*y* - *P*(*x*)|| *< ϵ* ）。这个 点的集合C称为共识，
* 保持并更新迄今为⽌建⽴的最⼤共识集合Cmax，直到 S 中的所有 3 种组合都被探索完毕。

最后， 它返回*C max* 它应该包含所有内点（并排除所有外点），并给我们固定的参数（公差*ϵ* ）。以下是我们方法的分辨异常点能力的一个例子（保留点为绿色）：

* + 1. 初始 (b) RANSAC “排序”后

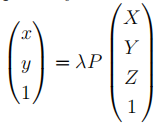
图 2.1: 有用点的选择

例如，在这里我们清楚地看到四个未选择的点不属于我们尝试重建的球轨迹，因为 这些是在弹跳 之前/之后捕获的好点！

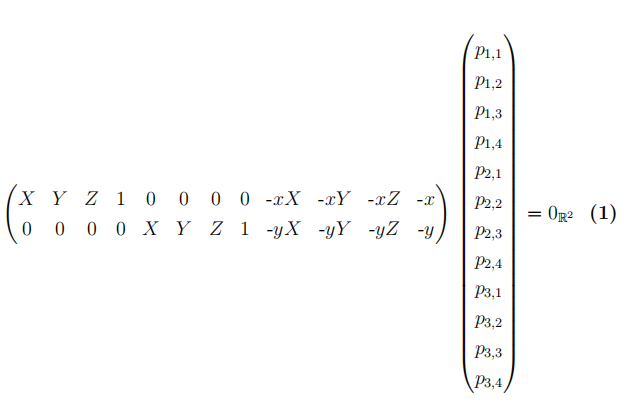
## 相机校准

相机校准作为 第一步，我们将重点聚焦在发现投影 矩阵 *P* ∈ R 3 *×* 4，将相关的[ *X, Y, Z* ]点与图像上的2D 映射[ *x, y* ]相关联起来的过程：

*Z* 

，直到任何⾮零标量因⼦ λ

将pij表⽰P 的系数，先前⼀对对应点之间的关系给出两个独⽴的⽅程，我们可以将其重写如下：

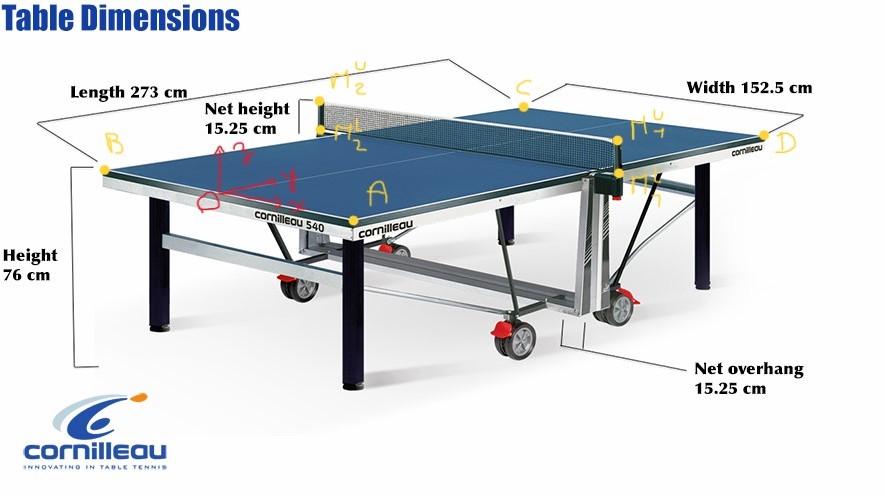


目标是找到 12 个*p i,j* 。因此，如果我们收集 *N*对对应的 2D 和 3D 点，其中N>=5，可提出至少 12 个方程，使得在系统上通过应用 DLT 算法，可找到所有系数：

*MP flatten* = 0R2 , 和 *M ∈ R*2*N×*12 可知。

然后，我们在表格图像上选择 N ≥ 5 个点，这些点的3D坐标在现实世界中是已知，给我们选择的点一个世界参考，⼀切都总结在下图中。

[经典：传统相机标定方法解析：直接线性法和Tsai两步标定法\_直接线性标定法](https://blog.csdn.net/jgwuhust/article/details/51485702)





数字 2.2： 相应的 点 萃取

最终获得的投影矩阵，可通过在已知的真实 3D 点上应⽤ P 来验证它“落在”图像上的位置，并与我们期望的位置进⾏⽐较。例如，我们尝试⽤我们的矩阵 P 映射桌⼦的中间、⽹的中上部分和每个桌⼦四分之⼀的中间，得到的映射“落在图像上”如下：



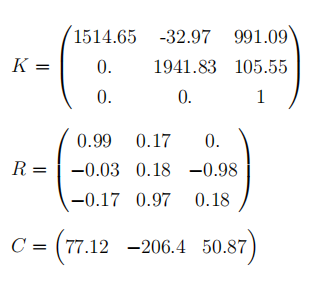
图2.3： 投影矩阵的⼀致性验证

它看起来连贯，且或多或少地保证了结果的质量。

有了*P* ，我们现在可以更进一步，对其进行“因式分解”以获得相机的内部参数（参数*K* ）和 外部参数（旋转矩阵*R* ，世界参考中的相机中心 *C*), 具体细节在NAYAR K共享的视频[3]中解释.

知道了 P = [M|m] = K[R|t]，我们可利⽤ M 的 QR 分解来找到 K、R，并利⽤ m =-MC 来找到 C

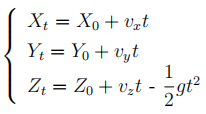
有一个大体了解（K 系数可能不会以绝对距离单位来解释），我们发现了以下内容：



## 利用物理原理提取球的真实运动

确定 P 后，下⼀步是利⽤该值和之前通过球跟踪（从单个⻆度）检索到的 2D 点序列及时间戳来恢复球的真实轨迹参数。Luca PIROTTI 在他的项⽬论⽂ [4] 中详细描述了这种⽅法，它依赖于⾃由运动物体的⼒学⽅程（具有初始条件）。

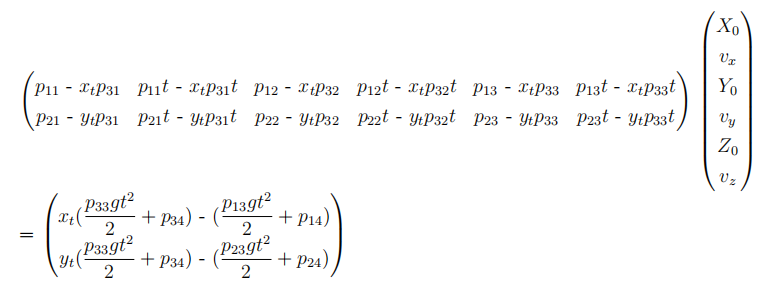
事实上，考虑单次击球，任意时刻 t 的球（真实）坐标都可以描述如下：

 **** 

其中 g = 981 cm/s2是重⼒加速度。

*Yt* = *Y*0 + *vyt*

这些运动⽅程确保（在单次击打期间）坐标变量[ X t , Y t , Z t ]是解耦的，并且彼此独立地演化，这带来了附加结果信息。 将[ X t , Y t , Z t ]替换为 其他的时间相关方程 mula (1)，将它们与相应的 2D 点[ x t , y t ]联系起来，我们可将后者重写为：



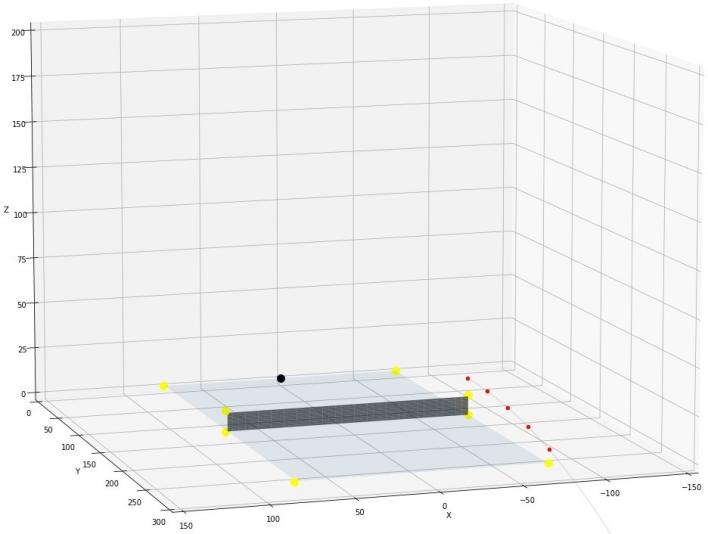
这导致我们得到⼀个 AX = B 形式的矩阵系统，X 是找到的未知向量，A 和 B 是已知元素（因为我们现在已经知道 P）。

使⽤ NumPy 线性代数库的最⼩⼆乘法，我们可以找到一个解决数组⽮量 X = (X0 Vx Y0 Vy Z0 Vz)T 最⼩化||AX-B||的解，因此给出真实轨迹参数的近似优化解决方案。

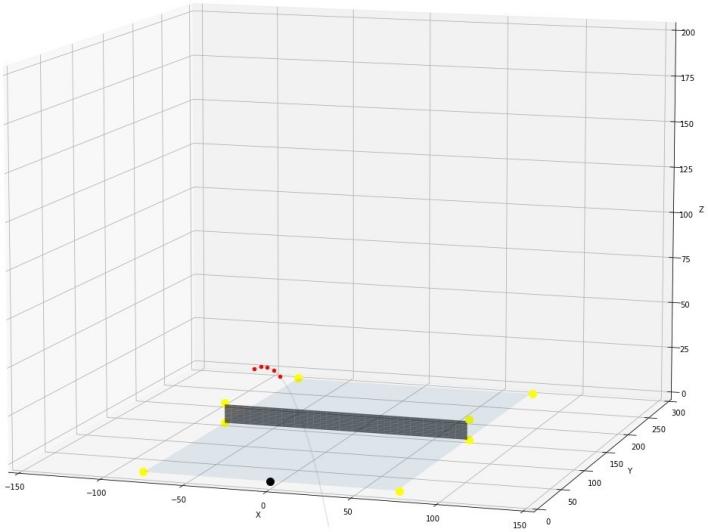
有了这些参数，我们现在可以随时计算球的真实轨迹（以及真实坐标），不要忘记它仅适⽤于单次shot，（如两次连续表面球接触之间的时间间隔）。取回真正的球在游戏的任何其他时间的位置，我们都需要另一个精心挑选的序列时间戳，来界定单个⾃由移动阶段！

## 结果讨论

以下是之前描述的体验的⼀些结果，以 3D 形式绘制。

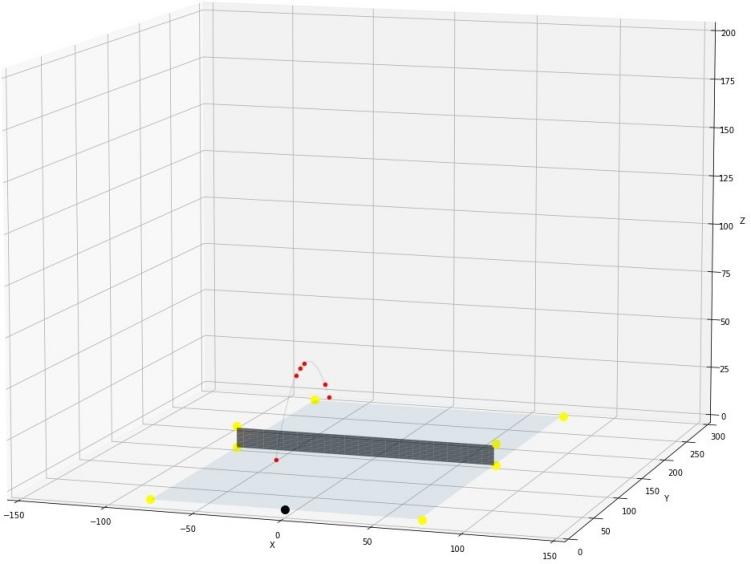


* + 1. (二)

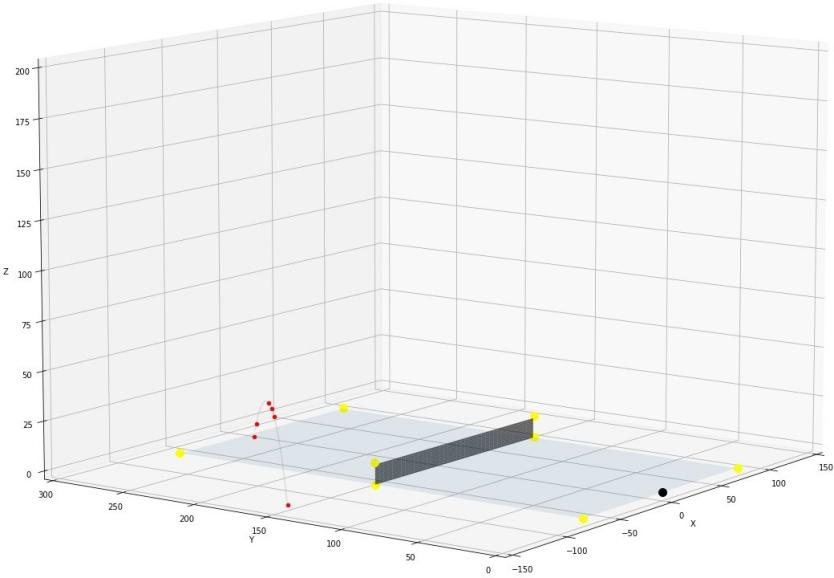


（C）

图 2.4： 重建 例子 1

* + - 1. (二)



（C）

图 2.5： 重建 例子 2

正如我们可以指出的，有时它或多或少表现得很好（参见示例 1），但有时所获得的结果与我们期望找到的事实相差甚远。 这模型，除了限于短序列上单个镜头的重建之外，有时表现很差。 这有点令人失望。考虑到所有累积/相乘的精度损失，整体处理的最终结果并不令人惊讶。

# 3 | 结论： 局限性和 可能的改进

通过这个项⽬，我们找到了⼀种从单⼀视⻆进行3D重建的⽅法，使⽤基于运动物理学的特定情境⽅法。然⽽，如前所述，该模型不是很稳健，实验结果有时令⼈失望。可以⽤很多因素来解释：

* 由于视频条件不佳，球跟踪不佳，因此我们很难达到很好的效果，并受到很⼤限制；
* 手动选择对应点可能会导致计算误差，并代⼊ P 的系数中
* RANSAC 算法清理输入序列不是很稳健：我们只能 在已经很短的输入序列上提取“正确”的点序列，并且通常涉及一些参数（宽容）调整，以做出 “令人满意” 的选择， 因此它降低管道的⾃动化程度
* 乒乓球服从如此简单的运动定律的假设可能过于乐观：它是⼀种⾮常特殊的球（轻且空），其运动⽅程可能应该包括物理⽅程中更复杂的元素，如空⽓阻⼒、摩擦⼒...

⼀些可能的经验和改进将留给下⼀个项⽬：

* 对于另⼀项试验，我们可以⾸先使⽤放置在桌⼦周围的多部智能⼿机从不同的⻆度拍摄每个游戏序列。因此，我们从⼀开始就不会受到很⼤的限制，我们可以应⽤更稳健的⽅法，如校准或未校准的三⻆测量。
* 我们可以尝试跟踪第⼀个球员球拍的⿊⾯，以便在任何时间和任何情况下跟踪两个⾯，然后跟踪球拍。
* 如果我们将来会遇到这样的另一个项目，但有一个多视角优势，我们也可以考虑在 3D 环境回放球员动作（例如 正手和反手）。

**4 | 参考**

1. 勃艮第大学， “图像分析：形态学和分类化” 课程， [https://ufrsciencestech.u-bourgogne.fr/master1/TraimentImages/partie\_MINIOT/cmTI.pdf](https://ufrsciencestech.u-bourgogne.fr/master1/TraitementImages/partie_MIGNIOT/cmTI.pdf)
2. SIMSEK Gökhan, “在图像和视频 使用 HSV 颜色 空间进行目标检测”， <https://blog.paralect.com/post/object-detection-image-and-video>
3. NAYAR K Shree，哥伦比亚大学，“相机校准”视频，   
   [https://www.](https://www.youtube.com/watch?v=S-UHiFsn-GI&list=PL2zRqk16wsdoCCLpou-dGo7QQNks1Ppzo) [youtube.com/watch?v=S-UHiFsn-GI&list=PL2zRqk16wsdoCCLpou-dGo7QQNks1Ppzo](https://www.youtube.com/watch?v=S-UHiFsn-GI&list=PL2zRqk16wsdoCCLpou-dGo7QQNks1Ppzo)
4. 皮罗塔 卢卡， 论文 “篮球视频上的单视图的球追踪和 3D 轨迹重建 ”, <https://www.politesi.polimi.it/handle/10589/142844>