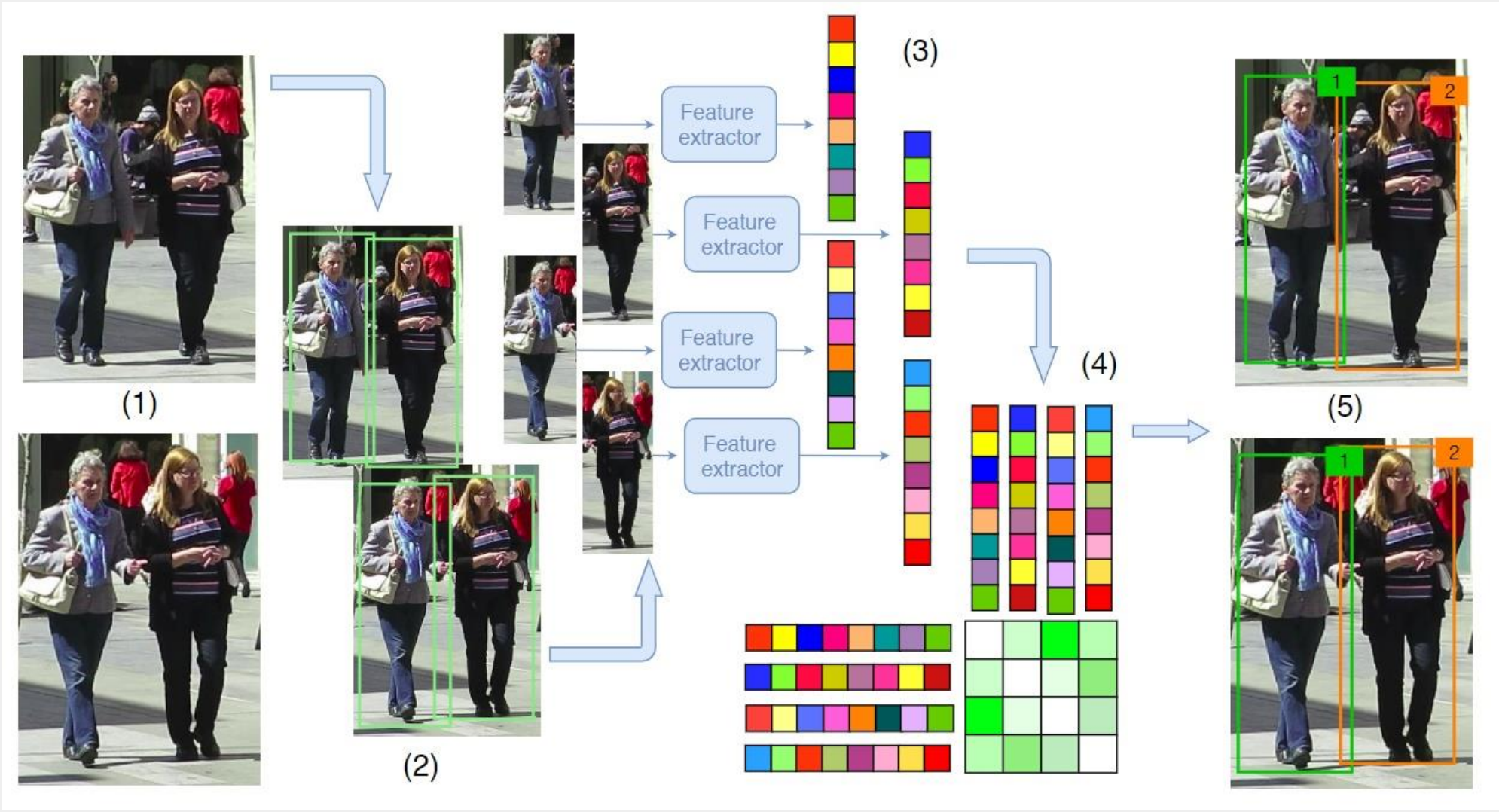
2020-07-27

组会报告

闫楠

算法框架：多目标跟踪MOT

1. ※检测边界框
2. 运动预测+视觉特征提取
3. ※数据关联/相似度计算
4. 分配ID完成跟踪



数据集：MOT Challenge为主，每年少量更新

评测指标：

- MOTA，检测漏报误报影响较大
- MOTP，检测框坐标精确程度的影响较大
- ID scores：针对目标ID，数据关联影响较大，考验跟踪算法效果

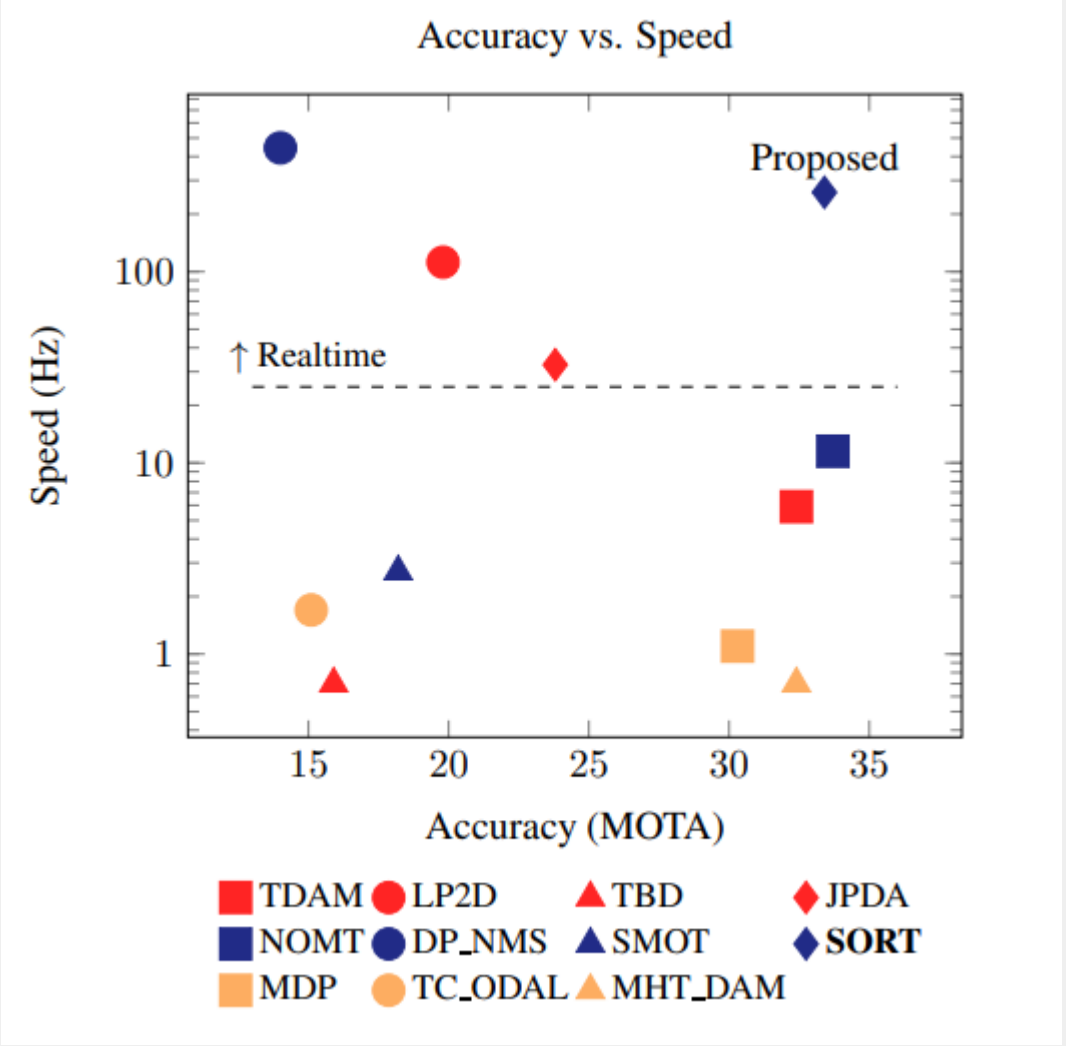
演示视频

SORT算法框架

Simple Online And Realtime Tracking

在线：只有当前帧和之前的信息
离线：整个视频的信息（有当前帧之后的）
实时：速度，跟踪比较追求实时的

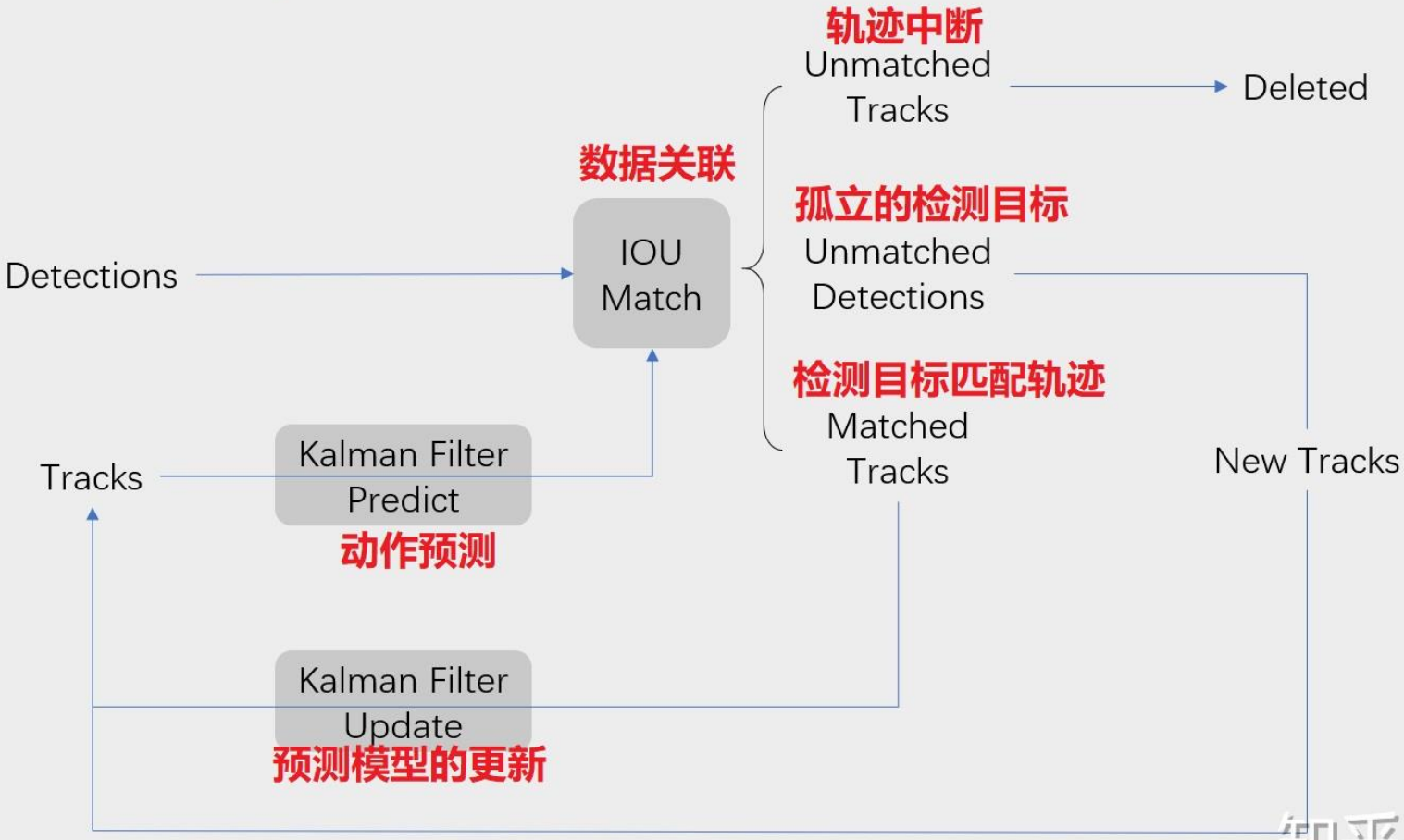
卡尔曼滤波+匈牙利算法，2016SOTA，260Hz，比SOTA快20倍



SORT算法框架

SORT (ICIP 2016)

Flow Chart



SORT算法框架

预测模型：
一帧中一个对象的运动状态表示为

$$\mathbf{x} = [u, v, s, r, \dot{u}, \dot{v}, \dot{s}]^T,$$

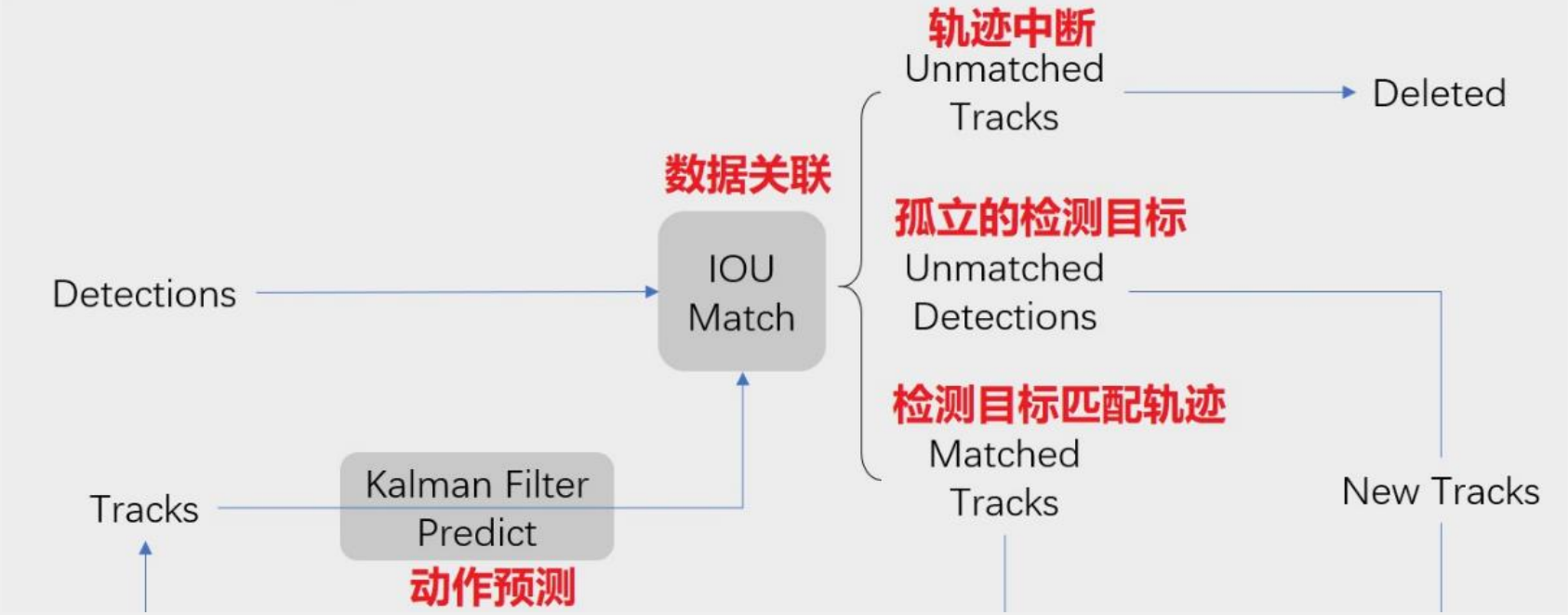
uvsr表示当前的位置和边界框，u'v's'是对下一帧的预测
假设u'v's'由uvsr线性表示——线性速度模型

卡尔曼滤波：优化估计算法， 用测量到的值估计不能直接测到的值。在输入数据受到噪声和误差影响的情况下，用它估计系统的状态和输出，比如位置、速度和方向。

得到下一时刻测量值后更新卡尔曼滤波的模型。

比如：u'v's'并不完全由uvsr线性表示，还有其他因素无法考虑。

SORT算法框架



数据关联：
轨迹预测结果+检测框，两两之间计算IoU，根据IoU用匈牙利算法计算关联程度。

SORT算法框架

指派问题：
有n项不同的任务，需要n个人分别完成其中的1项，每个人完成任务的时间不一样。于是就有一个问题，如何分配任务使得花费时间最少。

每个人完成每个任务耗时Cost矩阵，Cost总和最小
——每个检测框和每个轨迹预测之间的IoU矩阵，IoU总和最大

0-1规划问题，规划结果是一个n*n的矩阵，每行每列都只有一个1，其他是0，表示这行的人完成成这列的任务

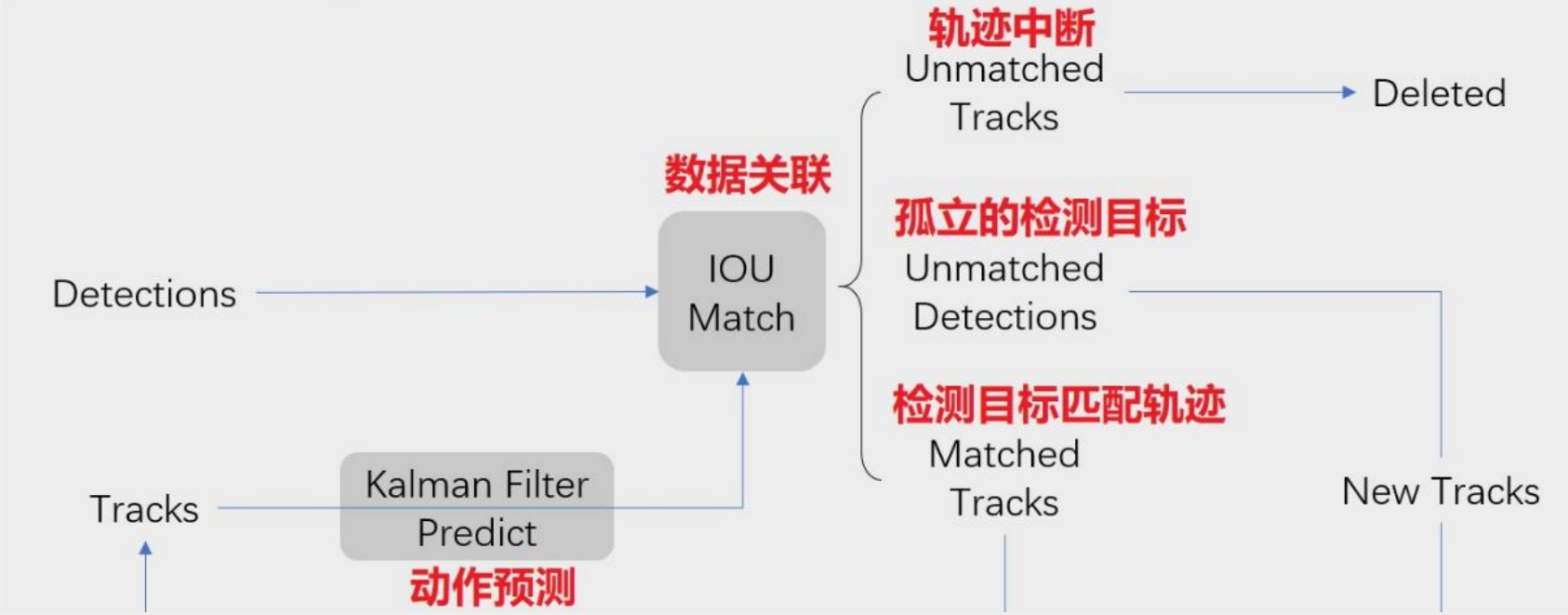
SORT算法框架

3.2 求解指派问题的匈牙利算法

由于指派问题的特殊性，又存在着由匈牙利数学家 Konig 提出的更为简便的解法—匈牙利算法。算法主要依据以下事实：如果系数矩阵 $C=(c_{ij})$ 一行（或一列）中每一元素都加上或减去同一个数，得到一个新矩阵 $B=(b_{ij})$，则以 C 或 B 为系数矩阵的指派问题具有相同的最优指派。

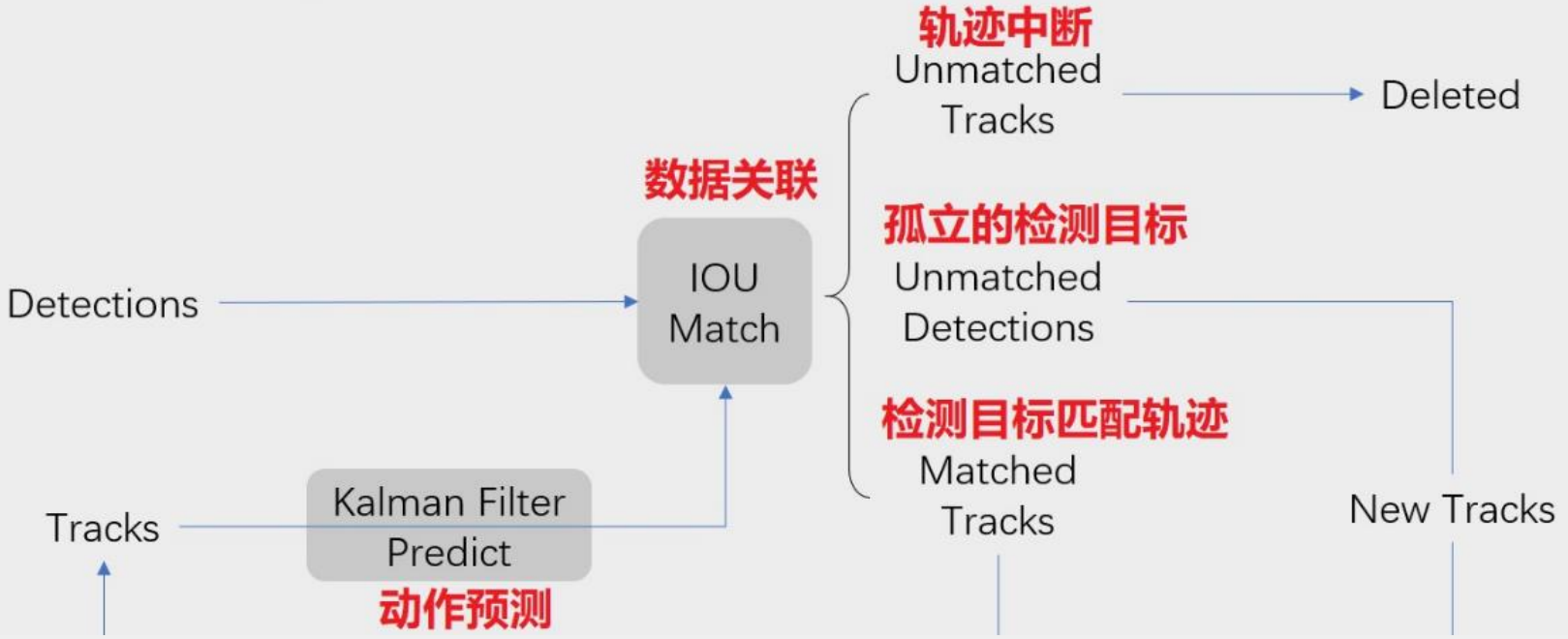
匈牙利解法：0-1规划的一种解法，通过矩阵变换，从系数矩阵变到只有n个1的n*n矩阵，即指派结果

SORT算法框架



不匹配轨迹（轨迹中断）：T时间后删除，可以是1，立刻删除。因为线性速度不合理，而SORT没有考虑重识别。这导致ID频繁变化。
不匹配的检测（孤立的检测目标）：创建新轨迹
匹配的轨迹（IoU大于阈值）：卡尔曼滤波更新

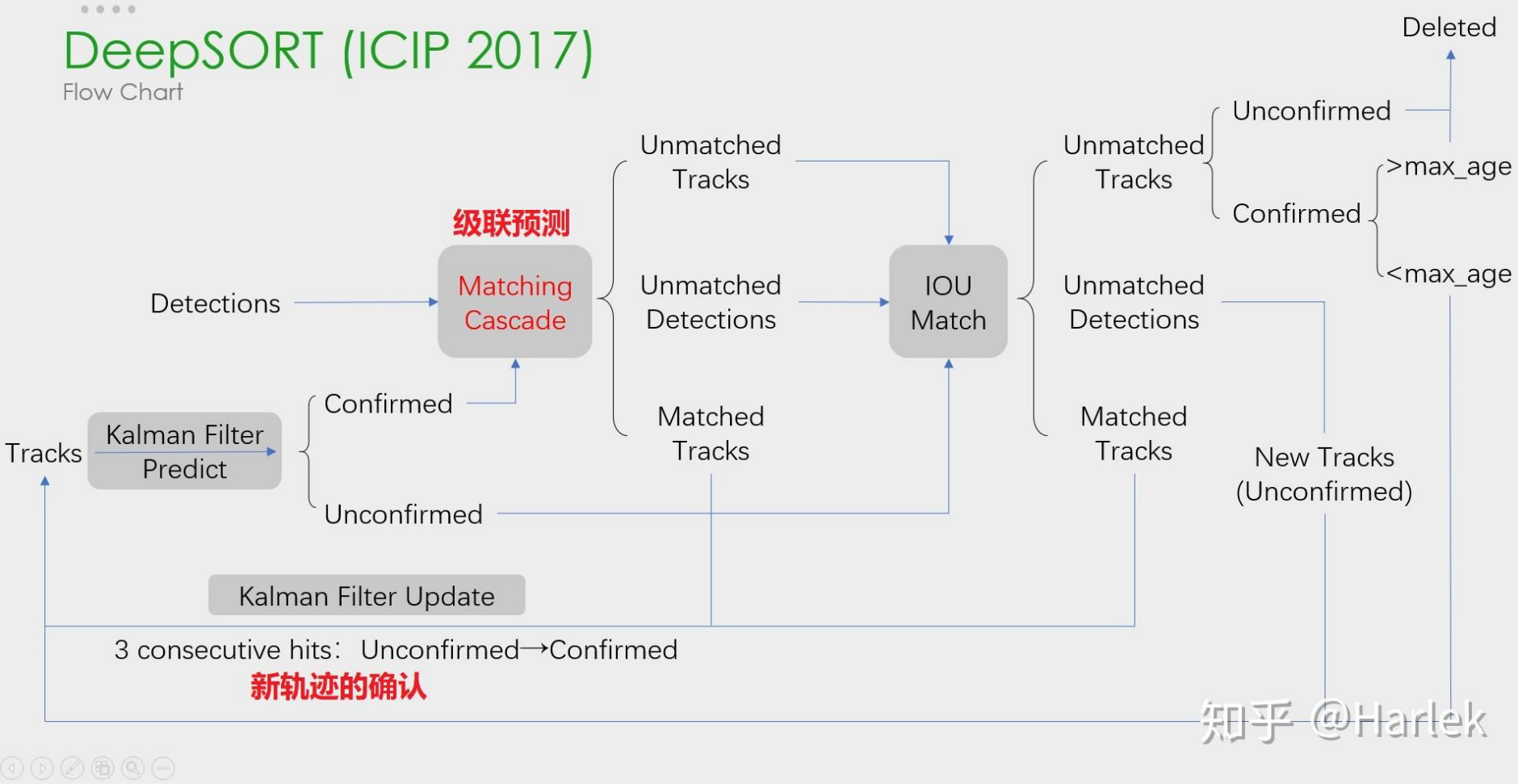
SORT算法框架



作者认为IoU可以解决短时遮挡的问题，因为障碍物和目标差不多大。但事实是大小差很多时这种办法无效，差不多大时误差很多。

演示视频ID=117

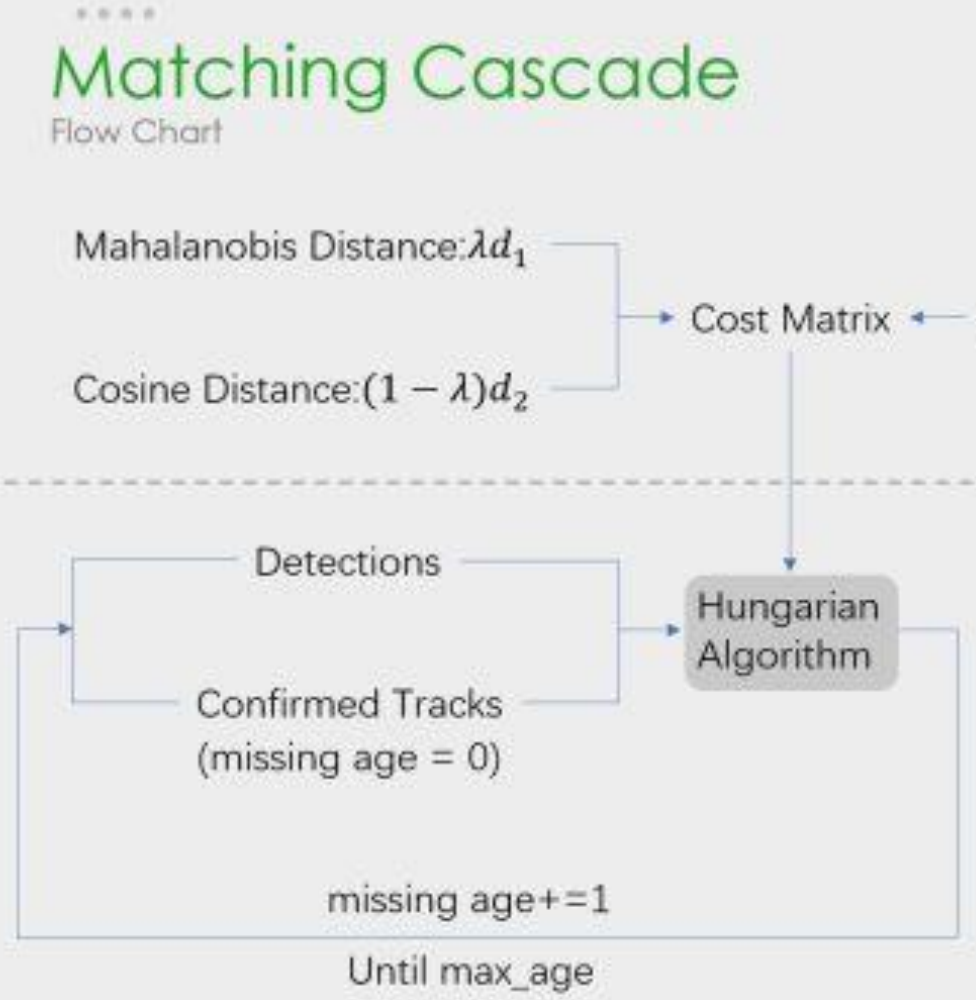
DeepSORT算法



预测模型加了一个速度，但也是线性

$$(u, v, r, h, x^*, y^*, r^*, h^*)$$

DeepSORT算法



Listing 1 Matching Cascade

Input: Track indices $\mathcal{T} = \{1, \ldots, N\}$, Detection indices $\mathcal{D} = \{1, \ldots, M\}$, Maximum age $A_{\max}$

Compute cost matrix $\boldsymbol{C} = [c_{i,j}]$ using Eq. 5
Compute gate matrix $\boldsymbol{B} = [b_{i,j}]$ using Eq. 6
Initialize set of matches $\mathcal{M} \leftarrow \emptyset$
Initialize set of unmatched detections $\mathcal{U} \leftarrow \mathcal{D}$
for $n \in \{1, \ldots, A_{\max}\}$ **do**
  Select tracks by age $\mathcal{T}_n \leftarrow \{i \in \mathcal{T} \mid a_i = n\}$
  $[x_{i,j}] \leftarrow$ min_cost_matching$(\boldsymbol{C}, \mathcal{T}_n, \mathcal{U})$
  $\mathcal{M} \leftarrow \mathcal{M} \cup \{(i,j) \mid b_{i,j} \cdot x_{i,j} > 0\}$
  $\mathcal{U} \leftarrow \mathcal{U} \setminus \{j \mid \sum_i b_{i,j} \cdot x_{i,j} > 0\}$
end for
return $\mathcal{M}, \mathcal{U}$

可改进点

1、one-stage：把重识别特征的CNN和检测CNN结合，就像RCNN曾经把ROI和特征提取结合一样，加快速度，但有可能失去一些精度

2、轨迹评分：级联匹配只降低失去匹配的轨迹的权重，而不增加长时间持续的轨迹的权重

3、用预测来弥补漏检

4、更好的运动模型：有时都设置马氏距离的权重为0，说明线性运动预测根本不管用，要换一个更复杂的非线性模型。

其他论文

Real-Time Multiple People Tracking With Deeply Learned Candidate Selection And Person Re-ID

MOTDT：用R-FCN加轨迹评分，给检测框和轨迹预测框做一个标准置信度，用置信度做非极大值抑制NMS，可以实现用预测弥补检测漏洞。

Towards Real-Time Multi-Object Tracking

JDE：检测和重识别特征集成为一个网络，用三元损失，还能自动学习这个三元损失。速度提升很多，并且没有牺牲精度。

其他论文

Deep Affinity Network for Multiple Object Tracking
DAN：端到端地学习目标的外观特征，包括对物体和他的周边环境，分层进行学习；还可以根据连续两帧的特征计算他们的关联性。

Tracking without bells and whistles
只使用目标检测方法+运动模型+RE-ID进行跟踪，不用在跟踪数据集上训练，直接用边界框来预测下一帧的位置，也能在不拥挤的场景下达到最先进的效果。

一个19年7月的综述：
“用深度学习来指导数据关联算法以及直接施展跟踪，这方面仍然是婴儿期，需要更多研究来明白深度算法在这里能否起作用。”

实验

官方的deepsort+yolo3用的库版本太久远了（16年），节约时间没有跑

试了两个deepsort+yolo5的开源代码，已经调试完，可以作为基准模型

熟悉了VOT数据集

计划

之前没思路还是读的太少，接下来加快速度，不精读了。

上面说的19年7月的综述还没读，下周读了，
再继续找些论文，比如比那篇综述晚的20年的，找思路。

关于用DL来解决数据关联，这确实很难，从这里下手要慎重。

提取视觉特征则比较有效，有可以从比如动作检测方面借鉴的方法、模型。

重识别特征可以找图像Re-ID的工作。

2020-07-27

组会报告

闫楠

THANK YOU

感谢聆听

PRESENTED BY OfficePLUS