ChatGPT

Multiple-Camera 3D Motion Tracking with Optical Flow

Background and Challenges

Multi-camera motion capture traditionally relies on detecting features (e.g. markers or body keypoints) in each view and triangulating them for 3D tracking [1] [2]. However, when these detectors fail – due to occlusion, motion blur, or lack of discernible features – optical flow can serve as a crucial signal to **maintain continuous tracking**. Using multiple calibrated monochrome cameras to estimate 3D trajectories offers distinct advantages: with more views, an object is seen from different angles, providing more information and robustness (e.g. one camera can take over if another loses sight) [3]. Indeed, researchers noted that "only recently people started to consider using multiple cameras to track a single object... having more information about the object" [3]. The key challenge is how to **turn 2D optical flow (image motion) into 3D motion estimates**, and how to maintain cross-view correspondences of points or features over time despite occlusions or detector dropouts.

Optical Flow as a Fallback in Multi-View Tracking

In multi-view human tracking systems, optical flow is often integrated as a fallback or complementary cue to handle failures of detection-based tracking. For example, Gamez et al. (2021) propose a multi-person pose tracker that uses optical flow to propagate human keypoints frame-to-frame under normal conditions, and switches to a predictive model (Kalman filter) only when the person becomes fully occluded [4] [5]. In their pipeline, each detected pose is associated over time by comparing it to a **flow-predicted pose** from the previous frame; if the person disappears in video (e.g. occluded or detector fails), the system falls back to a Kalman prediction until visual information returns [6] [5]. This demonstrates a common design: **optical-flow-based tracking** (which excels at short-term, fine motion continuity) is used whenever possible, and a more assumption-driven model (like constant-velocity Kalman) handles gaps when **optical flow cannot be obtained** [4] [5]. Such strategies exploit the fact that optical flow can track even **fast or unpredictable motions** of keypoints that a detector might miss, as long as the point remains visible [4]. Similar ideas appear in animal pose estimation (e.g. OptiFlex 2021), where initial deep keypoint predictions are refined by warping heatmaps with Lucas-Kanade optical flow, effectively using flow to enforce temporal consistency when the primary model is uncertain.

Multi-camera systems can extend these ideas by **combining flow across views**. A straightforward approach is to track features independently in each camera (e.g. using KLT point trackers or block-based motion trackers) and then **associate the tracks across cameras** using calibration. Eltoukhy et al. describe a two-camera system where each camera tracks an object's 2D motion, and a central module matches these trajectories via known camera geometry [7] [8]. By **triangulating** the 2D tracks, one obtains a 3D trajectory; if one camera loses the target (e.g. due to occlusion or leaving frame), another camera's view can continue to track it, and the system can seamlessly switch views [7]. Early work by Tsutsui et al. (2001) implemented an optical-flow-based person tracker with multiple cameras in an indoor setting, showing that multiple views can hand off the tracking and collectively maintain a person's trajectory even in cluttered scenes [9] [10]. These systems did not reconstruct full pose or shape, but rather followed the **overall 3D path** of the person by linking robust 2D motion cues. The correspondences between cameras were maintained either by stereo matching of the moving features or

by predictive geometric reasoning (e.g. if camera A's track predicts the person at a certain 3D location, search in camera B's image along the epipolar line for a matching motion) [11]. Maintaining consistent labeling of moving points across views can be done via **epipolar geometry constraints** and appearance consistency, sometimes assisted by a central fusion algorithm or even Bayesian filters that incorporate multi-view measurements.

From 2D Optical Flow to 3D Motion (Scene Flow)

When dense or per-point optical flow is available in each camera, a core question is how to lift that 2D motion into 3D. The computer vision literature has developed the concept of **scene flow**, which is essentially the 3D motion field corresponding to the observed 2D optical flow in multiple views [12]. One classic approach (Vedula et al., 1999; 2005) is to combine stereo and optical flow: at time t, a point's depth is known from multi-view stereo; optical flow tracks the point into time t+1 in each camera, and from these multi-view constraints one can solve for the 3D displacement vector of that point [13] [14]. For example, Li and Sclaroff (2005) fused stereo disparity with optical flow in a **coherent energy minimization framework**, enforcing brightness constancy across views and time. Their method handled the aperture problem by applying multi-scale adaptive smoothing on the flow fields, and produced a probabilistic estimate of disparity + flow that yielded reliable 3D motion (scene flow) even where traditional two-frame optical flow or stereo alone would be ambiguous [13] [14]. The result is a **dense 3D trajectory field** – essentially every visible point gets an estimated 3D velocity vector. Subsequent works refined this idea with global regularization: e.g. Vogel et al. (2011) and Park et al. (ECCV 2012) incorporate smoothness over the 3D flow field using techniques like tensor voting and piecewise rigidity. Park's system in particular leveraged **20 synchronized cameras** (outdoors) to first do multi-view stereo reconstruction of each frame, then computed initial 3D scene flow by **triangulating optical flows** between consecutive frames [15] [16]. They then applied a two-stage refinement (regularizing flow directions with local 3D smoothness, and magnitudes by ensuring consistency of the deformed point cloud) [17] [18]. This produced temporally consistent 3D trajectories of the scene points and could even reject outliers in the motion via robust tensor voting constraints [18]. In essence, these dense approaches maintain correspondences by enforcing that a point in 3D at frame t should project to points in each camera whose optical flow vectors agree with a single 3D motion; any inconsistent flows can be flagged as outliers. Such **scene flow** pipelines are often offline (due to heavy optimization), but optimized variants and hardware acceleration (e.g. using GPU flow estimation) have made it more feasible, especially for smaller regions of interest or real-time stereo pairs.

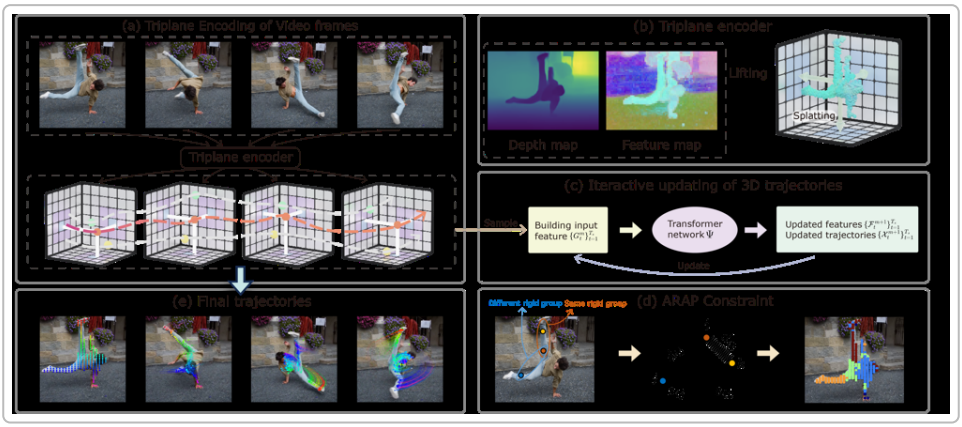

Figure: An example pipeline (SpatialTracker [19] [20], a recent research system) that lifts image features into 3D space for tracking. The model uses a learned depth (triplane) representation to track points in 3D with a transformer network, enforcing as-rigid-as-possible (ARAP) constraints on groups of points that move together [21] [22]. While this particular system is monocular (depth from a network), multi-camera

setups directly provide true depth, enabling similar 3D point trajectory estimation and grouping. Tracking in 3D (rather than separately per view) makes it easier to handle occlusion and complex motion, since the true spatial motion (e.g. rotations or out-of-plane moves) is represented explicitly [23] [24].

Not all systems need dense flow; **sparse feature tracking** across views can suffice for trajectory estimation. Some approaches focus on detecting a set of salient points on the object (or person) and tracking those over time with optical flow, then reconstructing their 3D positions at each frame. For instance, a multi-camera system might detect a person's outline or random visual features when a pose detector fails, and track those via optical flow. By **triangulating the new 2D positions** of those features, the system recovers an approximate 3D path even if the exact joints or shape are unknown. Robust estimation methods like RANSAC are often used here: for example, if an object is roughly rigid, one can fit a single 3D rigid-body motion (a rotation and translation) that best explains the 2D displacements of many feature points. This flow-to-rigid-fit strategy is common in model-based tracking of objects. Researchers have shown that optical flow cues alone can suffice to estimate 3D pose changes of rigid objects [25]. The idea is to use the optical flow vectors from multiple cameras to solve for the 6-degree-of-freedom motion that minimizes the reprojection error of all tracked points (often via least-squares or robust kernel). Because multiple views constrain the motion, even a simple **robust averaging** of per-view flow estimates can yield a stable 3D trajectory of the object's centroid or orientation. For non-rigid targets like humans, fully rigid fits won't apply globally, but parts of the body (e.g. torso) can be treated as approximately rigid over short time windows – an assumption used by some hybrid trackers to bridge brief failures in pose estimation.

Integrated Pipelines and Key Techniques

Combining optical flow with multi-view geometry requires tackling several technical points:

- **Maintaining Correspondences:** Multi-view systems must keep track of which point in camera A corresponds to which point in camera B and to the same physical point at the next time frame. This can be done by propagating known 3D points forward (project to each camera, then follow optical flow vectors to the new position in each view). Alternatively, as in some multi-target tracking systems, each camera produces 2D tracklets and the system matches these tracklets across cameras based on geometric consistency [26]. Methods like **epipolar alignment** (constraining that the motion of a point in one view should lie along the epipolar line of the motion seen in another) help prune incorrect correspondences. Particle filter approaches (Hadfield & Bowden 2013) even sample many hypothetical correspondences along epipolar lines – called scene particles – and weigh them by optical flow consistency and appearance [27]. By maintaining multiple hypotheses, they could handle long-term tracking without drifting to only one "best" guess, ultimately clustering these 3D particles into separate moving objects (e.g. two hands in a sign-language sequence) [28] [27]. This highlights a robust strategy: **don't rely on a single point match**; instead, propagate a distribution of possible matches and use redundancy across views and frames to identify consistent trajectories.

- **Sparse vs. Dense Flow:** There is a trade-off between tracking a few points and computing dense flow for all pixels. **Sparse flow (feature tracking)** is computationally cheaper and often more robust to noise (since it focuses on high-texture points), but it might miss large portions of the object if features are too few or temporarily lost. **Dense flow** provides complete motion information and can capture motion of any visible surface (useful for deformable objects or clothes), at the cost of solving a larger estimation problem. Some systems use sparse optical flow by default, and switch to dense flow analysis if the object's outline or surface needs to be recovered (for example, when a person turns such that fewer keypoints are visible, the system

might use dense flow on the person's silhouette or clothing pattern to estimate movement). Modern deep learning flow algorithms (like RAFT or transformer-based flows) can compute dense optical flow in near real-time [29] [30], making it feasible to incorporate dense cues even in live systems. Nonetheless, many real-time multi-camera trackers opt for a **mixed approach**: track a set of features (corners, learned keypoints) across frames, while also monitoring frame-to-frame optical flow in regions of interest to catch any motion the features might have missed. If a feature track fails, the system can spawn a new feature in that region using the dense flow information as a guide (essentially seeding a new track where movement was detected).

- **Rigid-body and Physically Inspired Constraints:** Introducing physical assumptions can greatly stabilize the 3D trajectory estimation. As noted, assuming local rigidity is useful – e.g. Furukawa and Ponce (2008) imposed a "local rigid motion" model for each mesh patch, coupled with a global non-rigid smoothing [31]. In practice, they tracked a dense mesh of a performer by first estimating the best rigid motion of each vertex's neighborhood (using optical flow and stereo to guide how that patch moved), then globally adjusting all vertices to be as consistent as possible with a plausible non-rigid deformation [31]. This two-stage optimization (local rigid + global regularization) allowed **complex non-rigid motions and occlusions to be handled effectively** in a markerless multi-camera setup [32]. Similarly, recent learning-based methods enforce constraints like the ARAP (as-rigid-as-possible) prior to group points moving together [33] [34], or constant velocity priors in Kalman filters for prediction during occlusion [6] [5]. **Robust estimation** techniques are crucial as well – outlier optical flows (from reflections, moving shadows, or background clutter) must be filtered. Many pipelines include RANSAC steps (e.g. when fitting a fundamental matrix or rigid transform to flow vectors) to drop inconsistent matches. Others use **robust M-estimators** in their energy formulations so that a few stray flow vectors don't warp the 3D estimate. For example, Li & Sclaroff's method incorporated uncertainty by generating probability distributions for flow and disparity, rather than committing to a single value [35] [36]. This way, uncertain regions (low texture, aperture problem areas) contributed less to the final 3D motion, improving reliability.

- **Real-Time vs Offline:** There is a spectrum from real-time systems (which often simplify the problem or leverage hardware) to offline high-precision systems. On the real-time end, an example is a multi-camera ball tracking system for sports, which might use optical flow to estimate velocity and a Kalman filter to triangulate 3D position at 50+ FPS. A 2004 system by Ren et al. achieved real-time 3D soccer ball tracking with multiple cameras, likely by combining simple background subtraction with optical flow prediction and multi-view triangulation (though relying on ball detections as well). In contrast, offline systems like dense scene flow or Furukawa's mesh tracker might take seconds or minutes per frame to compute, but yield a **rich motion capture** result (dense trajectories or a deforming mesh). The choice depends on the application: surveillance and robotics often need real-time, whereas biomechanics or film production can afford offline processing for higher fidelity. Hybrid approaches also exist: for instance, run a fast approximate tracker in real-time, but refine the trajectories later with batch optimization (smoothing the 3D paths, refining optical flow between key frames, etc.). Open-source projects like **OpenPose-based multi-view pipelines (e.g. Pose2Sim)** focus on accurate 3D pose output and currently do not integrate optical flow, but they could be extended with a flow module to fill in missing keypoints between frames [37] [38]. We are beginning to see research demos where multi-view neural networks output both poses and dense correspondence maps (flow or match descriptors), which will make building a robust system easier.

In summary, **multi-camera 3D trajectory estimation** can greatly benefit from optical flow integration. Optical flow (whether sparse or dense) provides the temporal glue that keeps tracking an object's motion when per-frame detection is unreliable. Multiple calibrated views provide the spatial context to turn

those 2D motions into 3D movements via triangulation and geometric consistency. Systems in literature have explored everything from early optical-flow fusion trackers [9] to modern learning-based scene flow estimators that recover full 3D motion fields [13] [14]. Key techniques include flow triangulation across views, epipolar-constrained feature tracking, local rigid-body motion fits, and global robust optimization to stitch together a coherent 3D trajectory or motion path. By using optical flow as a primary or fallback signal, multi-camera setups become more resilient – they can **continue tracking through momentary lapses of detection**, handle fast motions, and even capture non-rigid movements without needing explicit skeletal models. This makes such pipelines attractive for applications ranging from markerless human motion capture to object trajectory tracking in crowded or challenging environments. The ongoing research (including open-source efforts and CVPR 2024 highlights) suggests that combining multi-view geometry with optical flow will remain a rich direction for achieving **robust 3D motion tracking** in both real-time and high-precision offline contexts.

Sources: Multi-view markerless motion capture with optical flow [31]; Optical flow fusion for scene flow [13] [14]; Multi-person flow-based tracking [4] [5]; Early multi-cam flow tracking [9]; Recent 3D tracking advancements (SpatialTracker) [22] [24]; Multi-cam outdoor scene flow experiment [15], among others.

---

[1] [2] [13] [14] [31] [32] [35] [36] (PDF) Dense 3D Motion Capture from Synchronized Video Streams
https://www.researchgate.net/publication/221364919_Dense_3D_Motion_Capture_from_Synchronized_Video_Streams

[3] [7] [8] [11] Optical Flow and Feature Tracking
https://web.stanford.edu/class/ee392j/Winter2002/projects/eltoukhy_salama_report.pdf

[4] [5] [6] Multi-Person Pose Tracking with Occlusion Solving Using Motion Models
https://staff.aist.go.jp/e.yoshida/papers/GamezSII2021.pdf

[9] (PDF) Person Tracking Using Motion Detection and Optical Flow
https://www.researchgate.net/publication/27481793_Person_Tracking_Using_Motion_Detection_and_Optical_Flow

[10] Tracking unknown moving targets on omnidirectional vision
https://www.sciencedirect.com/science/article/pii/S004269890800566X

[12] [27] [28] personalpages.surrey.ac.uk
https://personalpages.surrey.ac.uk/s.hadfield/papers/Go%20With%20The%20Flow,
%20Hand%20Trajectories%20in%203D%20via%20Clustered%20Scene%20Flow.pdf

[15] [16] [17] [18] jaesik.info
https://jaesik.info/publications/data/12_eccv.pdf

[19] [20] SpatialTracker
https://henry123-boy.github.io/SpaTracker/

[21] [22] [23] [24] [29] [30] [33] [34] SpatialTracker: Tracking Any 2D Pixels in 3D Space
https://arxiv.org/html/2404.04319v1

[25] [PDF] Optical Flow-based 3D Human Motion Estimation from Monocular ...
https://graphics.tu-bs.de/upload/publications/alldieck2017optical.pdf

[26] [PDF] CS 585 Lecture on Multi-View Multi-Object Tracking
https://www.cs.bu.edu/faculty/betke/cs585/open/2024-cs585-multi-object-multi-view-tracking.pdf

[37] [38] GitHub - perfanalytics/pose2sim: Markerless kinematics with any cameras — From 2D Pose estimation to 3D OpenSim motion
https://github.com/perfanalytics/pose2sim