SLAM-Share: Visual Simultaneous Localization and Mapping (SLAM) for Real-time Multi-user Augmented Reality

Aditya Dhakal, Xukan Ran, Yunshu Wang, Jiasi Chen and K. K. Ramakrishnan

University of California, Riverside
Why is SLAM needed for AR?

- Augmented Reality (AR) applications must know the user device’s 3D location in the world.

- Simultaneous Localization And Mapping (SLAM) is the process for AR app to localize.

- SLAM is used when precision greater than GPS is desired.

---

Pokemon Go Buddy Adventure
Background: Visual SLAM Execution Steps

- Visual SLAM is based on images of environment
  1. **Features are extracted** from image frame
  2. **Tracking:** Extracted features are compared to existing map to localize
  3. **Mapping:** New features are inserted into the map
     1. **Map-points:** feature points that will go in the map
     2. **Keyframe:** Image frame and its position and orientation
  4. Error is minimized in the map

- We base SLAM-Share on ORB-SLAM, a Visual SLAM application

Map

Trajectory
Multi-user AR Requires Information Sharing

User (A) - Edge Server - User (B)
How does latency affect the AR display?

User B’s View

User A’s View (Ground Truth)

Case (a): Without information sharing, no holograms appear.

Case (b): With slow tracking, holograms may appear later.

Case (c): With slow map merging, holograms may appear inaccurately placed.
**Problem:** Tracking is slow on mobile clients! < 30 FPS

**Problem:** Multiple clients’ maps need to be merged quickly! Default: takes 3 sec

**Our contributions:** New offloading architecture with IMU assist, GPU assist, map merging, and shared memory for high-throughput, multi-user visual SLAM for AR
Tracking

Mapping

Map merge

Shared memory

Tracking

GPU assist

Tracking

GPU assist

Tracking

GPU assist

IMU assist

IMU assist

IMU assist

AR client

AR client

AR client
**GPU assist:**
How does the GPU help?

- Search Local Points is time-consuming → SLAM-Share exploits parallel threads

- ORB-Extraction is time-consuming → SLAM-Share exploits GPU parallelism

- Overall, SLAM-Share reduces tracking time by more than 40% compared to ORB-SLAM3 run in CPU only

![Diagram showing time comparison between ORB-SLAM3 (OS3) and SLAM-Share (S-Sh) for different datasets (KITTI, V202) and configurations (Stereo, Mono). The bars indicate the time in milliseconds for each task: Search Local Points, Pose-Prediction, ORB-Matching, ORB-Extraction. The line graph at the bottom right highlights the requirement.]
IMU assist

- Once server contact restored
  - Client merges IMU + SLAM pose

- Evaluation: IMU-based tracking is accurate for a short time
  - But long term IMU-based tracking accumulates errors

<table>
<thead>
<tr>
<th>RTT (ms)</th>
<th>IMU-Tracking region ATE RMSE (cm)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 (Baseline)</td>
<td>2.41</td>
</tr>
<tr>
<td>90</td>
<td>2.45</td>
</tr>
<tr>
<td>200</td>
<td>2.67</td>
</tr>
<tr>
<td>300</td>
<td>2.71</td>
</tr>
<tr>
<td>10000</td>
<td>300</td>
</tr>
</tbody>
</table>

Server Contact Lost!
Mapping

- Map merge
- Shared memory

- Tracking
  - GPU assist

- IMU assist

- AR client
Why is map merging needed?

- Map merging fuses the shared information between users
  - Map merging brings together users’ maps and puts them in same “perspective”

- Without map merging, the virtual objects will be misplaced for some users
- With map merging, the virtual objects are at the same place for all users
Map Merge Example

Each client keeps local copies of shared map → inefficient! → Shared memory for global map
Does ATE remain low throughout?

- We show a scenario of merging 3 clients’ maps with SLAM-Share

- Need low ATE for accurate virtual object placement
**How Fast Does SLAM-Share Merge Maps?**

- **Baseline**: multi-user implementation of Edge-SLAM

- Baseline map transfer from client to Edge server adds latency

- SLAM-Share’s use of shared memory lowers overheads

- Merging new map to global map is time consuming
  - SLAM-Share incrementally updates the map

---

<table>
<thead>
<tr>
<th>Component</th>
<th>Baseline (ms)</th>
<th>SLAM-Share (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Serialization (app)</td>
<td>78.1</td>
<td>N/A</td>
</tr>
<tr>
<td>Encoding</td>
<td>N/A</td>
<td>3</td>
</tr>
<tr>
<td>Map transfer (to server)</td>
<td>66</td>
<td>0.11</td>
</tr>
<tr>
<td>Deserialization (app)</td>
<td>390.8</td>
<td>0</td>
</tr>
<tr>
<td>Map Merging</td>
<td>2339</td>
<td>190</td>
</tr>
<tr>
<td>Map transfer (to client)</td>
<td>6.4</td>
<td>0.1</td>
</tr>
<tr>
<td>Load Map (in client)</td>
<td>19.8</td>
<td>N/A</td>
</tr>
<tr>
<td>Total</td>
<td>2900.1</td>
<td>193.21</td>
</tr>
</tbody>
</table>

SLAM-Share Map Merge is an order of magnitude faster

---

Is Multi-User SLAM-Share as good as Single User ORB-SLAM3?

- Evaluation
  - ATE of map created by SLAM-Share with 9.4 Mbit/second bandwidth between client and Edge
  - ATE of the map created by SLAM-Share with 300ms delay added for each packet

→ SLAM-Share multi-user maps are as accurate as those of single-user ORB-SLAM3
Is SLAM-Share Accurate When There is Network Delay?

- Comparisons
  - SLAM-Share and baseline
  - With and without added delay

→ SLAM-Share has almost same accuracy despite 300 ms delay

→ Baseline suffers from higher short-term inaccuracies with increased delay
CPU Overhead of SLAM-Share vs. Baseline Clients

- We evaluated the overall CPU use in SLAM-Share and Baseline clients

- SLAM-Share uses less than 1% of single CPU Core
Conclusion

- SLAM-Share improves key components of Visual SLAM: tracking and mapping
  - Intelligently re-thinks partitioning of SLAM tasks between mobile client and the Edge Cloud

- SLAM-Share exploits GPU-based tracking on the edge cloud
  - Speed up of tracking by more than 40%

- SLAM-Share uses shared-memory on edge cloud to rapidly merge client maps
  - SLAM-Share’s Map Merging is an order of magnitude faster

- SLAM-Share achieves high-throughput multi-user visual SLAM-Share
  - Very resource/power efficient on client - very small CPU and memory consumption

- Open-source code available: https://github.com/network-lab2/slam-share