Ranged Kinematics and Time of Flight Sensors for Shared Autonomy Robotic Systems

Michael Gleicher
Department of Computer Sciences
University of Wisconsin Madison
Human Graphics Interaction
authoring pictures, videos, animations

Human Robot Interaction
robots!

Human Data Interaction
visualization, visual analytics, interactive learning

Michael Gleicher
Visual Computing Group
Acknowledgments

New Stuff (RangedIK, Sensors)
Main Students:
Carter Sifferman
Yeping Wang

Key Collaborator:
Mohit Gupta

Old Stuff
Main students/post-doc:
Daniel Rakita (prof @ Yale)
Mike Hagenow (post-doc @ MIT)
Pragathi Praveena (post-doc @ CMU)
Emanuel Senft (research leader @ IDIAP)

And many others…

Main Collaborators:
Bilge Mutlu, Mike Zinn
Ranged Kinematics and Time of Flight Sensors for Shared Autonomy Robotic Systems

Michael Gleicher
Department of Computer Sciences
University of Wisconsin Madison
### Shared Autonomy

Mixing direct (manual) control and automation

Our old stuff (manual tele-op) moving to new things

Motivation

---

### Ranged Kinematics

Exploit task tolerances to get better control

Provides robustness and responsiveness in motion synthesis

New motion methods

---

### Time of Flight Sensors

Use cheap sensors when and where we need them

Allows algorithms to see and respond to dynamic situations

Methods for new sensors
The last Robotics Talk...

Can we can enable (smart) people to work with dumb robots?

Robot intelligence may be **overrated** less important in places where we can effectively use human intelligence and skill

If we can successfully exploit the person to help the robot.
Folding a Shirt

The robot doesn’t do a great job... but this was 2016

And it’s hard!
You need to see the shirt and understand its geometry
You need to understand the tasks and goals
You need to know how cloth reacts to being pushed and pulled
You need to predict how the shirt will move
You need to choose where to grab and which way to pull things
You need to plan/strategize on how to get the shirt folded
What happened?

To fold a shirt...
You need to see the shirt and understand its geometry
You need to understand the tasks and goals
You need to know how cloth reacts to being pushed and pulled
You need to predict how the shirt will move
You need to choose where to grab and which way to pull things
You need to plan/strategize on how to get the shirt folded
What happened?

To fold a shirt...
You need to see the shirt and understand its geometry
You need to understand the tasks and goals
You need to know how cloth reacts to being pushed and pulled
You need to predict how the shirt will move
You need to choose where to grab and which way to pull things
You need to plan/strategize on how to get the shirt folded

This robot didn’t do any of that

The human did the “hard parts”
What made this work?

Responsive Kinematics
Mimicry control:
It “feels” like using your hand
Responsiveness over accuracy
You can always adjust
Details only matter sometimes

Good Awareness (visibility)
You can see what you are doing
You can figure what to do
You can respond to anything
You can always adjust
Mimicry: Direct Control Tele-Operation

Making the robot do what you do.


Mimicry-based Teleoperation
Approach Overview

Hand 6-DOF Space → Spatial Mapping → End Effector 6-DOF Space → Inverse Kinematics → Robot Joint Angle Space
Perceptual Hack

When moving fast...
• Smoothness (rough position) is important
• Orientation is less important
• Maintain manipulability (so can adjust when slow down)

When moving slow...
• Pose is important (position / orientation)
• Manipulability is more important (so can adjust if needed)
Not Traditional Inverse Kinematics

Position and orientation goals are not the only goals we don’t need precise matches!

Other things are important too:
  smoothness / continuity
  responsiveness
  avoid self-collisions and singularities

Make tradeoffs (multi-objective optimization)
Solution must...

Important:
• Maintain responsiveness
• Be low latency
• Afford direct control
• Run at high frequency
• Avoid self collisions
• Avoid kinematic singularities
• Produce smooth motions without jumps

Less important:
Accurately match position and orientation
Preserve Manipulability

\[ C(\theta) := \sqrt{|\det(J(\theta)J(\theta)^T)|} - s_{\min} \geq 0 \]
Keep away from bad things

Keep manipulability above minimum (away from singularities)
Keep distance to self-collision above minimum

Efficient formulations as constraints

Use fast approximation for the self-collision distance function.
Relaxed IK

IK (position/orientation matching) is just one goal

Self-collision and singularity avoidance are priorities

Everything else is a tradeoff
  be flexible (allow for different objectives)
  be dynamic (tune weights responsively)

And do it fast...
Objectives (weighted)

Position matching
Orientation matching
Joint velocities (smoothness)
End-effector velocity (control)

Far from problems

Constraints

Maintain manipulability
Avoid self-collisions
(later: environment collisions)
How do we balance competing objectives?

Use constraints for high-priority Objectives (costs) for everything else.  \( c_i(q) \)

Hard to balance weights for different objectives
different scales
different falloffs

Use shaping functions!
\[
s_i \in \mathcal{R} \rightarrow \mathcal{R}
\]
\[
\sum_i s_i \left( c_i(q) \right)
\]
Shaping functions

\[ s_i \in \mathcal{R} \rightarrow \mathcal{R} \]
Awareness

How to provide a good view to the operator?
  Remote (via video)
  Even local operator (can’t be too close)

Exploit flexibility of RelaxedIK Framework
  Optimize camera as well as end effector


Motion Retargeting Optimization

User Motion Input

Live Video Stream

Motion Optimization

Manipulation Robot Configuration (per update)

Camera Robot Configuration (per update)

Camera Robot Motion Optimization
Camera Distance

Camera should move out for context when user moves robot quickly
Organize Pills
Even better camera control

Let the camera robot see (look for AR tag on gripper)
  avoid occlusions
  should always be able to see hand and “goal”

Optimize both robots together
  keep manipulation visible

User control
  searching and exploring, nudging, control over distance
Acknowledgements

Students
Danny Rakita (Yale)
Pragathi Praveena (CMU)
Guru Subramani (Intuitive)
Mike Hagenow (MIT)

Co-PIs
Bilge Mutlu
Michael Zinn

Post-Doc
Emanuel Senft (IDIAP)
Beyond Tele-Op

Too much work for the user!
But we still want them to be involved

Shared Autonomy
Some manual control
Some automation
From Direct Control to Shared Autonomy

Responsive Kinematics
Algorithms can’t get stuck
Still need flexibility to react
Smooth motion for viewer
  (interpretability)
Need even more flexibility
Need to limit imprecision
  (user can’t reason/adjust)

Awareness
Algorithms must be aware
Avoid hard to see things
Need sensing
RangedIK: IK for Ranged-Goal Tasks


RangedIK: IK for Ranged-Goal Tasks

Idea: give ranges for goals (not poses)
ranges in position and orientation

Adds degrees of freedom to problem (like extra joints)
Allows for solutions within those bounds
Ranged Tasks

**Good for algorithmic control**
Specify what is needed
insures “good enough”
More flexibility
simple algorithms work
Better motions
flexibility

**Good for tele-operation**
Only control important things
less to worry about
More flexibility
even more responsive
Better motions
robot is more responsive
Challenges of Ranged Tasks

How to specify – in a manner that fits optimization framework

How to solve – in a manner that preserves the good stuff

Can people control it?
  or will they feel out of control?
  (because they are sharing control)
Specify

Goals as “objectives” (functions)

Combine with type of relationships
• Specific (value) goals
• Preferred value (but range is OK)
• Anywhere in range is OK
• Completely don’t care
Example 1: Wiping (Eraser)

Eraser is radially symmetric - don’t care about orientation
Example 2: Writing

Pen can rotate around its axis (symmetric)
Pen can tilt (up to 30 degrees)
This really helps... *(more precise, less arm motion)*

**Specific 6dof Goals**

**Allow rotational Freedoms**
Example 3: Spraying

Allow some wiggle parallel
Example 4: Filling a glass

Cup is symmetric
Position doesn’t have to be exact
How to implement it?

Must fit into RelaxedIK framework
- objectives that can be easily combined
- objectives that are easy to solve (good derivatives)

Use shaping functions (on objectives)
- Not barrier objectives (too hard to solve)
- Not constraints (too hard to make tradeoffs)

Keep “large bowls” to attract to solution
Change behavior when close
Loss Functions

Large bowls (attract toward goals)
Steep boundaries (keep inside)
Experiments

Four tasks (wiping, writing, spraying, filling a glass)
Provided example paths
Measure precision on specific goals
All approaches stay within the tolerances (on non-goal variables)

Three approaches
  RangedIK
  RelaxedIK – not as smooth, not as precise
  TrackIK (exact) – very precise, but infeasible results
What happens if the user tries to control it?

User controls important degrees of freedom
System uses other degrees of freedom (within range) to meet other goals (smoothness, responsiveness, …)

Do users feel out of control?

Does better responsiveness make up for less control?
**Exact Mimicry**

- Robot mimics all 6 DoFs
- Users have full control
- May lose manipulability

**Functional Mimicry**

- Robot controls rotation of the cup
- Accurate, smooth, feasible motions
- User’s control less *direct*
Human-Subject Experiment

H: Autonomous robot adjustments within task tolerances in mimicry-based telemanipulation will lead to better task performance and user experience.
Within-participants design
Condition order was counter-balanced
20 participants
Results

Objective Measures (Lower is Better)
The functional mimicry manipulator that exploited task tolerances was perceived to be more under control, predictable, fluent, and trustworthy.
Ranged IK

Allow freedom in some DoFs

Upsides

more precision in DoFs we care about
better smoothness and manipulability (for path following)
better objective metrics (for tele-op)
better subjective metrics (users feel more in control!)
Now on to sensing...
Using Small, Cheap Time of Flight Sensors (SPADs)


Fangzhou Mu(o), Carter Sifferman(s), Sacha Jungerman(o), Yiquan Li(o), Mark Han(o), Michael Gleicher, Mohit Gupta, and Yin Li. 2024. Towards 3D Vision with Low-Cost Single-Photon Cameras. *CVPR 2024*. https://doi.org/10.48550/arXiv.2403.17801

Carter Sifferman(s), William Sun(o), Mohit Gupta, and Michael Gleicher. 2024. Using a Proximity Sensor to Detect Deviations in a Planar Surface. *(submitted for publication)*
A Vision...
Strategic Sensing

Put sensors where we need them!

They need to be:
Small (low size/weight)
Cheap (low cost)
Cheap (low power)
Cheap (low computation)
Cheap (low bandwidth)
We’re not the only ones…

Escobedo et al., 2021

Crazyflie 2.1 with z-ranger deck

Miniature ToF SPADs

AMS TMF8820

Breakout Board

Sensor

ST VL6180X  ST VL53L8CH

Earpiece Proximity Sensors
Time-Resolved Proximity Sensors

AMS TMF8820

ST VL6180X

ST VL53L4CX

1cm – 5m

Typical Range

<10mW/measurement

Typical Power Consumption
What can you buy?

2565 Results for "spad"

Product: SparkFun Qwiic dToF Imager - TMF8820
Price: $20.95

Description
The TMF8820 is a direct time-of-flight (dToF) sensor in a single modular package with associated VCSEL. The dToF device is based on SiPd, TDC and histogram technology and achieves 500 mm detection range. Due to its low scattering, it supports 3x multizone output data and a wide, dynamically adjustable, field of view. A multi-ansi-array (MLA) inside the package above the VCSEL, widens up the FoV (field of Illumination). All processing of the raw data is performed on-chip and the TMF8820 provides distance information together with confidence values on its I2C interface.

Details

Features
- Direct ToF technology with high-sensitivity SPAD detection
- 3x3 multi-zone configuration with multi-object detection
- Adjustable field of view (up to 65° diagonal)
- Fast Time-to-Digital Converter (TDC) architecture
- Sub-nano second light pulse
- 10 - 5000mm distance sensing @ 30Hz
- On-chip histogram processing
- 940nm VCSEL, Class 1 Eye Safety
- High-performance on-chip sunlight rejection filter and algorithm
- Small modular OLOA 2.0mm x 4.6mm x 1.4mm package

Benefits
- Delivers high SNR, wide dynamic range and no multi-path reflections
- Enables dark and sunlight environment distance measurement within ±5%
- Adaptable dToF for custom scene matching
- Provides best-in-class resolution ranging mode detection sensing
- Enables accurate distance measurements
- Provides high accuracy, greater distance between cover glass, dynamic cover glass calibration, dirt or smudge removal and crossstalk compensation
- Eye-safety circuitry (VCSEL driver: Class 1, fault occurs)
- Optical filters w/algorithm support enables high ambient light resilience
- Reduced board space requirements, enables low-profile system designs in restricted space industrial designs

Parameters
- Sensor zones: Multi
- Number of detection zones: 9
- Detection distance: Max. 500 cm
- Water level optical filters: Yes
- Field of view: 65°
- Operating ambient temperature: Min. -40° C, Max. 85° C
- Supply voltage: Min. 2.7 V, Max. 3.5 V, 5.0 V
- Package: CGL4A12
- Interface: Input/output
- Package dimension: L 14.6 mm, W 2.0 mm, H 1.4 mm
What are they selling you?

4.5cm
What do you get?

3.7 cm
Is the “laser-beam model” good for anything?

Yes, actually...
Geometric Calibration of Single-Pixel Distance Sensors

**Problem:**
where is sensor on robot?

**Given:**
Scene has a plane
Known robot motion

**Recover:**
Pose of sensor and plane relative to robot coordinates
What do you get?

3.7 cm
What are they selling you?

3x3 Pixels

Pixels are still finite areas!

Per-Pixel Distance Estimates

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>24</td>
<td>22</td>
<td>23</td>
</tr>
<tr>
<td>18</td>
<td>19</td>
<td>19</td>
</tr>
<tr>
<td>17</td>
<td>16</td>
<td>15</td>
</tr>
</tbody>
</table>
What is going on inside?

Send out pulses

Lots of photons

Sample the ones that come back

Quantity and time
Measure when things return

Record times of each return

Single Photon Avalanche Diode

SPAD
Count many different photons
Accumulate over time
Sampling of all photons sent
Many photons...
Each with a time stamp
Count “time bins” (discretize)
Transient Histograms!

Each bin corresponds to time range
Count how many photons return
Time = distance
Background: ToF SPADs

Light Source

Field-of-Illumination

Sensor

Outgoing laser pulse

Returning laser pulse (transient)

Quantized returning laser pulse (transient histogram)
Miniature ToF SPADs – AMS TMF8820

3x3 Pixels

Per-Pixel Histograms

Internal Proprietary algorithms

Per-Pixel Distance Estimates
The opportunity...
Robotics Applications of Transient Histograms

We get more information! (histogram, not a number)
- We understand what the numbers are
- We can use the whole histogram

But...
- It’s a new kind of information (depth statistics)
- There are ambiguities
- We need new approaches!
The Problem(s) ...

Amount of light returned (for a bin) depends on...

1. The amount of stuff (at that distance)
2. The reflectance of that stuff (at that distance)
3. Systematic problems (e.g., cross-talk, pile-up, ...)
4. Noise (randomness)
The Problem(s) ...

Amount of light returned (for a bin) depends on...

1. The amount of stuff (at that distance)
2. The reflectance of that stuff (at that distance)
3. Systematic problems (e.g., cross-talk, pile-up, ...)
4. Noise (randomness)

**Fundamental Problem**
(we’ll come back to it)

Can’t tell where photon came from
Can’t distinguish reflectance from amount
The Problems(s) ...

Amount of light returned (for a bin) depends on...

1. The amount of stuff (at that distance)
2. The reflectance of that stuff (at that distance)
3. Systematic problems (e.g., cross-talk, pile-up, ...)
4. Noise (randomness)
SNR Demo

White chipboard (about 1 mm)
2cm square on the background

White on white, no texture

Clearly and reliably measured

(the catch is coming up)
The Problems(s) ...

Amount of light returned (for a bin) depends on...

1. The amount of stuff (at that distance)
2. The reflectance of that stuff (at that distance)
3. Systematic problems (e.g., cross-talk, pile-up, ...)
4. Noise (randomness)

These devices have surprisingly good Signal-to-Noise Ratios! (SNR)
The Problems(s) ...

Amount of light returned (for a bin) depends on...

1. The amount of stuff (at that distance)
2. The reflectance of that stuff (at that distance)
3. **Systematic problems** (e.g., cross-talk, pile-up, ...)
4. Noise (randomness)
Can you really use the histograms?

Idea: pick a simple (but practical problem), dive deep

Explore different paths…

1. Use measurements from sensor (internal algorithm)
2. Use simple approaches to histogram
3. Use data+heuristics to exploit histograms (task knowledge)
4. Use a really good model of the sensor and parameter fitting
   Differentiable Rendering, Render-and-Compare
Unlocking the Performance of Proximity Sensors by Utilizing Transient Histograms

RA-L / ICRA 2024

Carter Sifferman1, Yeping Wang1, Mohit Gupta1, and Michael Gleicher1

Abstract—We provide methods which recover planar scene geometry by utilizing the transient histograms captured by a class of close-range time-of-flight (ToF) distance sensor. A transient histogram is a one-dimensional temporal waveform which encodes the arrival time of photons incident on the ToF sensor. Typically, a sensor processes the transient histogram using a proprietary algorithm to produce distance estimates, which are commonly used in several robotics applications. Our methods utilize the transient histograms directly to enable recovery of planar geometry more accurately than is possible using only proprietary distance estimates, and consistent recovery of the albedo of the planar surface, which is not possible with proprietary distance estimates alone. This is accomplished via a differentiable rendering pipeline, which simulates the transient imaging process, allowing direct optimization of scene geometry to match observations. To validate our methods, we capture 3,800 measurements of eight planar surfaces from a wide range of viewpoints, and show that our method outperforms the proprietary-distance-estimate baseline by an order of magnitude in most scenarios. We demonstrate a simple robotics application which uses our method to sense the distance to and slope of a planar surface from a sensor mounted on the end effector of a robot arm.

1. INTRODUCTION

Of PTICAL time-of-flight proximity sensors which measure scene transients have recently become widely available. These sensors operate by illuminating the scene with a pulse of light, and measuring the shape of that pulse over time as it returns back from the scene in a transient histogram, as shown in Figure 1. These transient histgrams have seen use in robotics due to their ability to reliably report a distance estimate over a wide range (1cm - 5m) while being small (< 20 mm³), lightweight, and low-power (on the order of milliwatts per measurement) [1], [2]. Because of their form factor, transient sensors can be placed in locations where higher resolution 3D sensors cannot, such as on the gripper or links of a robot manipulator, or on very small robots. While these sensors have many desirable properties, existing robotics applications do not utilize the transient histograms, instead relying on low-resolution (at most

Manuscript received: May 31, 2023; Revised August 19, 2023; Accepted August 23, 2023.

This paper was recommended for publication by Editor Tamim Asif upon evaluation of the Associate Editor and Reviewers’ comments. This work was supported by Los Alamos National Laboratory and the Department of Energy, a University of Wisconsin Vilas Award, and National Science Foundation awards 1551727, 1943158, 2075069, and 2007125. 2023 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works. The authors are with the Department of Computer Sciences, University of Wisconsin-Madison, Madison 53706, USA. (Sifferman/yeping/mohit/mgleicher@cs.wisc.edu).
Can We Recover Parameters of a Plane?

Plane finding is a simple starting point with some practical use cases:
- Drone landing
- Pick-and-place with a robot arm
Naïve Method

Distance Estimates

Project and Fit Plane

3DoF Planar Parameters

<table>
<thead>
<tr>
<th>24</th>
<th>22</th>
<th>23</th>
</tr>
</thead>
<tbody>
<tr>
<td>18</td>
<td>19</td>
<td>19</td>
</tr>
<tr>
<td>17</td>
<td>16</td>
<td>15</td>
</tr>
</tbody>
</table>
Evaluation

Eight materials with varying reflectance properties
Distance: 1-30cm
Angles-of-incidence: 0° - 30°
3,800 measurements total
Method 2: Simple Histogram Use

1. First full bin (space carving)
2. Simple peak finding (max bin)
3. Simple interpolation

None of these perform very well compared to ...
Method 3: Peak Finding

Hueristic: we know it’s a plane, so we expect one big peak
Data-driven: tune for correction factors (weights to adjust locations)

Spline fitting to precisely localize the first big peak
## Method 3: Peak Finding Results

<table>
<thead>
<tr>
<th>Method</th>
<th>Angular Error (°)</th>
<th>Linear Error (mm)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Mean</td>
<td>Median</td>
</tr>
<tr>
<td>Peak Finding - Calibrated</td>
<td>3.57</td>
<td>2.22</td>
</tr>
<tr>
<td>Peak Finding - Uncalibrated</td>
<td>5.68</td>
<td>3.87</td>
</tr>
<tr>
<td>Proprietary Distances - Calibrated</td>
<td>7.34</td>
<td>4.71</td>
</tr>
<tr>
<td>Proprietary Distances - Uncalibrated</td>
<td>8.87</td>
<td>4.71</td>
</tr>
</tbody>
</table>
Method 3: Pitfalls

We are still distilling the histogram to a single value, which throws out information.

Does not utilize the magnitude of the histogram, which could be used to recover the albedo of the surface.
Method 4: Differentiable Render-and-Compare

**Idea:** render what we expect the sensor to see, compare it to what the sensor actually sees, and optimize unknown parameters to make the render match the actual measurement.

Render Function: $R(G, F, C, \delta)$

- Planar Geometry
- Reflectance Parameters
- Sensor Intrinsics
- Outgoing Laser Impulse
Method 4: Differentiable Render-and-Compare

Scene Parameters
(AoI, Azimuth, Albedo, Specularity, Ambient Light...)

Calibrated Sensor Intrinsic
(FoV, Pulse Count, Light Intensity...)

Sample Random Rays in FoV
Ray-Plane Intersection
Times-of-flight and Per-Ray Intensities

Differentiable Binning Weighted by Intensity
Idealized Scene Response

Gradients w.r.t. inputs
Automatic Differentiation

Outgoing Laser Impulse
Captured Histogram

Loss Function

Rendered Histogram
The key: Need a good forward model

Build a model by:
1. Modeling sensor phenomena
2. Using known data to fit and tune parameters
3. Leaving unknowns (e.g., surface properties) as variables
The Sensor Model

1. Phong surface reflection (provides surface parameters)
2. SPAD saturation (non-linearity between light and count)
3. Histogram formulation
4. Laser impulse (not uniform over pixel)
5. Cross-talk
## Method 4: Results

More results (wider range of plane geometry, varying surfaces) in the paper!

<table>
<thead>
<tr>
<th>Method</th>
<th>Angular Error (°)</th>
<th>Linear Error (mm)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Mean</td>
<td>Median</td>
</tr>
<tr>
<td>Differentiable Rendering</td>
<td>3.40</td>
<td>1.97</td>
</tr>
<tr>
<td>Peak Finding - Calibrated</td>
<td>3.57</td>
<td>2.22</td>
</tr>
<tr>
<td>Peak Finding - Naive</td>
<td>5.68</td>
<td>3.87</td>
</tr>
<tr>
<td>Proprietary Distances - Calibrated</td>
<td>7.34</td>
<td>4.71</td>
</tr>
<tr>
<td>Proprietary Distances - Naive</td>
<td>8.87</td>
<td>4.71</td>
</tr>
</tbody>
</table>
Method 4: Albedo Recovery

Measurements are from a variety of distances and angles-of-incidence.
The model gets better...

That paper was May 2023....

This year (CVPR 2024)...

Towards 3D Vision with Low-Cost Single-Photon Cameras

Fangzhou Mu, Carter Siffman, Sacha Jungerman, Yiquan Li, Mark Han, Michael Gleicher, Mohit Gupta, Yin Li

1 co-first author, 2 equal contributions

University of Wisconsin-Madison

Figure 1. We demonstrate that measurements from spatially distributed low-cost single-photon proximity sensors (left) can be used to reconstruct 3D shape of real world objects (right). Our method combines a differentiable image formation model and neural rendering to recover 3D geometry based on measurements (transient histograms) from sensors with known poses. This is done by minimizing the difference between the observed and rendered sensor measurements. For clarity, a subset of sensor poses and measurements are shown.

Abstract

We present a method for reconstructing 3D shape of arbitrary Lambertian objects based on measurements by miniature, energy-efficient, low-cost single-photon cameras. These cameras, operating at time resolved image sensors, illuminate the scene with a very fast pulse of diffuse light and record the shape of that pulse as it returns back from the scene at a high temporal resolution. We propose to model this image formation process, account for its non-idealities, and adapt neural rendering to reconstruct 3D geometry from a set of spatially distributed sensors with known poses. We show that our approach can successfully recover complex 3D shapes from simulated data. We further demonstrate 3D object reconstruction from real-world captures, utilizing measurements from a commodity proximity sensor. Our work draws a connection between image-based modeling and active range scanning, and offers a step towards 3D vision with single-photon cameras. Our project webpage is at https://cpsaff.github.io/towards_3d_vision/.

1. Introduction

Reconstructing 3D shape of real objects remains a central problem in vision, solutions to which have evolved into two parallel branches. Image-based modeling [45] leverages a plethora of visual cues from multiple photographs (e.g., stereo, motion, shading), leading to problems including multi-view stereo [45], photometric stereo [1] and the more recent neural radiation fields (NeRFs) [35]. Conversely, active range scanning [21] combines an active light source with an imaging sensor, giving rise to imaging techniques such as structured light [13], and time-of-flight [10]. Conventional wisdom suggests that range scanning yields more precise 3D geometry than image-based modeling at the cost of using specialized, expensive hardware. An emerging approach for range scanning is direct time-of-flight imaging with active single-photon cameras, a form of time-resolved image sensor. This approach couples a pico-to-nanosecond detector with a fast coherent light source, illuminates the scene with a very short pulse of light, and measures the intensity of the light over time as it reflects back from the scene. The resulting incident wavefront is recorded and quantized, forming a transient histogram. A special case of this approach is single-photon LiDAR, in which the light source (laser) is highly focused, the detector finds the peak in the histogram, and the sensor reports a single-distance value per detector pixel. When using a diffuse light source, these time-resolved sensors capture visual information beyond the distance measurements extracted in LiDAR. Transient histograms in this case record distributions of times-of-flight, encoding the product of scene geometry and reflectance over each imaged scene patch [21]. The entirety of information in the transient histogram has been previously utilized in applications such as fluorescence
More than a plane?

Use known “camera” positions (many)
Use a rich and complex model (neural radiance/occupancy field)
The Problems(s) ...

Amount of light returned (for a bin) depends on...

1. The amount of stuff (at that distance)
2. The reflectance of that stuff (at that distance)
3. Systematic problems (e.g., cross-talk, pile-up, ...)
4. Noise (randomness)

Careful modeling of the sensor can resolve many of the problems
The Problems(s) ...

Amount of light returned (for a bin) depends on...

1. The amount of stuff (at that distance)
2. The reflectance of that stuff (at that distance)
3. Systematic problems (e.g., cross-talk, pile-up,...)
4. Noise (randomness)

---

**Fundamental Problem**

Can’t tell where photon came from
Can’t distinguish reflectance from amount
What does this histogram bin mean?

There is something reflective in this bin
N photons came back
N Photons from this region

Where in the region did they bounce from?
What could it be?
Reflectance matters

Reflectance * area

Roughly uniform illumination

Many photons absorbed or bounced in other directions
Why does this matter?

(I said this picture would return)

If we know what these things are, we can be very sensitive
An application: Deviation Detection

Detect small things on the floor

A bump or a separate object?

Holes, cliffs, ...

Is the floor really flat?

Abstract—We investigate methods for determining if a planar surface contains geometric deviations (e.g. protrusions, objects, divots, or cliffs) using only an instantaneous measurement from a miniature optical time-of-flight proximity sensor. The key to our method is to take advantage of raw time-of-flight data captured by the-off-the-shelf commodity proximity sensors. We provide an analysis of the problem in which we identify the key ambiguity between geometry and surface photometricities. To overcome this challenge, we propose fitting a Gaussian mixture model to a small dataset of planar surface measurements. This model implicitly captures the expected geometry and distribution of photometrics of the planar surface, and is used to identify measurements which are likely to contain deviations. We characterize our method on a variety of surfaces and planar deviations, and find that our method utilizing raw time-of-flight data outperforms baselines which use only derived proximity estimates across a range of scenarios. We build an example application in which our method enables mobile robot obstacle and cliff avoidance over a wide field-of-view.

I. INTRODUCTION

Optical time-of-flight proximity sensors are widely used in robotics to sense the distance to nearby objects for tasks such as obstacle avoidance [1] or localization [2], [3]. These sensors are low-cost, low-power, and are available in low-resolution (e.g. 4x4 pixel) arrays requiring minimal data bandwidth. The proximity estimates reported by these sensors are over summaries over a wide (e.g. 10°) field-of-view per-pixel, which is advantageous for some applications (e.g. conservative obstacle avoidance), but means the sensors are ineffective at detecting small geometric deviations. In this work, we show that this limitation can be overcome by utilizing readily available raw time-of-flight information captured by these sensors. With this data, we are able to detect geometric deviations in a planar surface with more accuracy than is possible with distance estimates alone.

Our method utilizes raw time-of-flight data captured by consumer-grade time resolved active time-of-flight sensors. These sensors operate by illuminating a wide (30°) patch of the scene with a pulse of light, and capturing the intensity of light over time as it bounces back from the scene in a 1D temporal waveform called a transient histogram [4], [5]. These sensors are available for less than $5 USD and are widely used in robotics applications [6], [7]. In addition to their low cost, they are very small (<20 mm) and low-power (<10 milliwatts per measurement) [8], [9]. Typical applications do not utilize the transient histogram captured by these sensors, instead relying on a proprietary algorithm onboard the sensor to extract a single distance estimate per pixel. While this estimate is convenient for many tasks, it is not ideal for many others, as it obscures relevant information about the scene which is encoded in the shape and magnitude of the transient histogram.

In this work, we aim to detect geometric deviations on planar surfaces (e.g. objects, divots, cliffs, or walls). We assume that the position of the sensor relative to the planar surface is fixed, and require a small dataset of measurements of a flat planar surface from that fixed position. Our method operates on a per-frame basis (i.e. does not rely on sensor motion). We do not aim to detect the exact nature of the planar deviation, but rather classify a measurement as planar or not. Detecting planar deviations under these conditions is useful for robotics applications like mobile robot and drone navigation. This capability could also be useful for e.g. safely landing a drone on a flat and level surface, or using a robot manipulator to safely place a cup of liquid on a clear portion of a tabletop.

Our method enables detection of deviations in a very small size, weight, and compute footprint, making it particularly useful for resource-constrained scenarios like micro-drones, and for distributed sensing with sensors mounted at many points on a larger robot.

The key contributions of this work are: 1) an analysis of...
Ambiguous!

Reflectance variation and bumps can look the same!
Ambiguities
Ambiguities
In practice...

Surfaces have very big differences

Spread out over the entire space

Deviations are (relatively) small
What to do?

Model the background (surface) using Gaussian Mixture Models

Build a model of a library of surfaces
Assume robot starts in a clear patch and build a specific model

Out of distribution implies deviation
It works...

<table>
<thead>
<tr>
<th>Method</th>
<th>Overall AUROC</th>
<th>Bottle Cap</th>
<th>Cable</th>
<th>Chair</th>
<th>Fork</th>
<th>Glove</th>
<th>SD Card</th>
<th>Tennis Ball</th>
<th>Wall</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ours (Histograms)</td>
<td>0.84</td>
<td>0.83</td>
<td>0.87</td>
<td>0.80</td>
<td>0.84</td>
<td>0.87</td>
<td>0.76</td>
<td>0.86</td>
<td>0.89</td>
</tr>
<tr>
<td>Histogram Peaks</td>
<td>0.63</td>
<td>0.61</td>
<td>0.65</td>
<td>0.61</td>
<td>0.63</td>
<td>0.66</td>
<td>0.57</td>
<td>0.71</td>
<td>0.62</td>
</tr>
<tr>
<td>Proprietary Distances</td>
<td>0.60</td>
<td>0.58</td>
<td>0.55</td>
<td>0.59</td>
<td>0.54</td>
<td>0.63</td>
<td>0.70</td>
<td>0.52</td>
<td>0.68</td>
</tr>
</tbody>
</table>
It even works on a robot

This is 3 – we are planning a full ring (8)
The parts are cheap enough that we can do this
Summary:
We can get a lot out of cheap sensors!

Use inexpensive SPAD sensors

Make use of internal representations
   Transient Histograms

Sensor modeling to correctly interpret measurements
Algorithmic choices to perform tasks (given ambiguities)
What's Next?

Simple sensors
Use motion to disambiguate
Use multiple sensors
More applications (tasks)

Strategic Sensing:
How do we “right size” our sensing (not have too much) by putting sensors where we need them?

Relaxed and Ranged Kinematics
Understand tradeoffs
Algorithms for new tasks
Sampling strategies

Imprecise Robotics:
How do we “right size” our motion synthesis to get good enough answers (rather than exact/optimal ones)?
Shared Autonomy?

We need robustness
We need sensing
(and many other things too)
Thanks!
To you for listening.
To my students and collaborators.
To the NSF, NASA and Los Alamos for funding.

Ranged Kinematics and
Time of Flight Sensors
for Shared Autonomy Robotic Systems

Michael Gleicher
University of Wisconsin Madison
gleicherc@cs.wisc.edu