Mozart: A Mobile ToF System for Sensing in the Dark through Phase Manipulation

Zhiyuan Xie†
The Chinese University of Hong Kong
Hong Kong SAR, China
xavier_ie@link.cuhk.edu.hk

Xiaomin Ouyang‡
The Chinese University of Hong Kong
Hong Kong SAR, China
xmouyang@link.cuhk.edu.hk

Li Pan
The Chinese University of Hong Kong
Hong Kong SAR, China
lipan@link.cuhk.edu.hk

Wenrui Lu
University of Michigan, Ann Arbor
Ann Arbor, MI, USA
wenruilu@umich.edu

Guoliang Xing†
The Chinese University of Hong Kong
Hong Kong SAR, China
glxing@link.cuhk.edu.hk

Xiaoming Liu
Michigan State University
East Lansing, MI, USA
liuxm@cse.msu.edu

ABSTRACT
Sensing in low-light and dark environments has a wide range of applications. However, existing sensing technologies suffer several major challenges, such as excessive noise and low resolution. This paper proposes Mozart - a new mobile sensing system that leverages off-the-shelf Time-of-Flight (ToF) depth cameras to generate high-resolution and rich-in-texture maps for applications in dark scenarios. The design of Mozart is based on our key observation that the phase components of ToF measurements can be manipulated to expose texture information. Through in-depth analysis of the physical reflection model, we show that the textures can be exposed and enhanced using highly compute-efficient phase manipulation functions. By exploiting the physics texture models, we propose an autoencoder-based unsupervised learning approach that can automatically learn efficient representations from phase components to generate high-resolution maps. We implemented Mozart on several Android smartphone models1, and an edge testbed with standalone ToF camera platforms for various applications in the dark. The results show that Mozart can work in real time and delivers significant improvement over existing sensing technologies. Therefore, Mozart offers a low-cost, high-performance sensing technology for next-generation applications in the dark.

CCS CONCEPTS
• Hardware → Displays and imagers; • Computing methodologies → Appearance and texture representations; • Computer systems organization → Embedded and cyber-physical systems.

1 A demo video of Mozart smartphone App working in the dark is available at https://youtu.be/qBEJXVH_8.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

MobiSys ’23, June 18–22, 2023, Helsinki, Finland
© 2023 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 978-1-4503-9007-0/23/06...$15.00
https://doi.org/10.1145/3581791.3595940

KEYWORDS
ToF depth camera, Sensing in the dark, Phase manipulation

1 INTRODUCTION
Sensing in low-light and dark environments has a wide range of applications, such as smart building, smart health, and robot navigation. For example, a highly desirable feature of smart door locks is automatic unlock via face recognition or secret hand gestures in dark environments [31, 43]. Moreover, many health monitoring systems and human activity recognition systems [52, 54, 62] require 24/7 sensing capabilities, e.g., detecting Sudden Infant Death Syndrome (SIDS) during sleep using a smart baby monitor [63].

As summarized in Table 1, although there already exist many sensing technologies that function to some extent in dark conditions, they cannot meet the requirements of high-resolution sensing applications. RF-based systems such as mmWave radar and Wi-Fi are not interfered with by visible light. Unfortunately, their sensing data are highly sparse [46, 58, 72], making them poorly suited for applications that require high-resolution results such as human faces and hand gestures. Thermal, IR, and depth cameras can work in the dark. However, thermal cameras have limited resolution [28]. Although IR cameras can provide more detailed information, they rely on strong IR emissions, which incur high power consumption ranging from 5 to 20W [15, 61]. Moreover, off-the-shelf commercial IR cameras suffer from the over-exposure effect when objects are too close and can only capture image details within a short distance (typically up to 5m [71]) due to the fast decay of light intensity [65]. ToF depth cameras have a more extended range and lower power consumption and are increasingly embedded in smartphones or used as standalone sensors for 3D applications. However, by design, ToF depth cameras cannot capture most texture information of the scene [66].

In this paper, we proposed Mozart, a novel sensing system that leverages off-the-shelf ToF cameras to generate high-resolution and rich-in-texture maps for dark scenarios. As shown in Figure 1, Mozart maps can significantly augment the performance of various sensing tasks in the dark. The design of Mozart is based on our key observation that the phase components of ToF measurements can be carefully controlled (which we refer to as “phase manipulation”) to generate high-resolution maps with detailed textures [67]. To design Mozart, we first present an in-depth analysis of the physical
reflection model for exposing texture information through phase manipulation. Our key finding is that the textures can be exposed and enhanced using highly compute-efficient phase manipulation functions. Lastly, we propose an end-to-end autoencoder-based unsupervised learning approach to automatically learn efficient representations from the phase component maps to generate Mozart maps. To train the deep autoencoder, we design three novel loss functions by exploiting the physics models we proposed, including the albedo similarity loss, the illumination attenuation loss, and the uniform distribution loss. Our approach offers several key advantages, including requiring no labeled training data and being highly scalable in different applications without manual system tuning.

We implement Mozart on several Android smartphone models and mainstream standalone ToF cameras. The results show that Mozart maps can be generated in real time on smartphones due to the extremely low overhead. Moreover, Mozart can generate high-resolution maps at about 23 frames per second on edge computing platforms, and is compatible with mainstream ToF cameras. We evaluate the performance of Mozart using three new datasets we collected in low-light and dark conditions, including human tracking, face recognition, and gesture recognition, which involves a total of 33 subjects and contains over 1,000,000 data frames. The results show that Mozart maps outperform all baseline modalities (including RGB, IR, and mmWave Radar) in dark environments. For example, in gesture recognition, Mozart outperforms RGB images, mmWave Radar, and depth images by 93.4%, 88.46%, and 45.76%, respectively. Moreover, compared with IR maps, Mozart delivers a performance improvement of up to 29.1% with a substantially smaller variance.

Our key contributions are summarized as follows:

• To the best of our knowledge, Mozart is the first low-cost, high-performance sensing system to generate high-resolution maps using a single off-the-shelf ToF camera in low-light and dark environments.

• We provide an in-depth analysis of the physics models for exposing and enhancing textures. Our key finding is that the textures can be generated using highly lightweight phase manipulation functions.

• By exploiting the physics texture models, we propose a deep autoencoder-based texture generation approach that can automatically learn efficient representations from phase maps to generate Mozart maps.

• We implement Mozart on several smartphone models as well as edge platforms with mainstream ToF modules. Our evaluation using three self-collected datasets in the dark shows that Mozart outperforms existing baselines and can work in real time.

2 RELATED WORK

Sensing Technologies for Applications in the Dark. Sensing in the dark has a wide range of applications such as robot navigation, face authentication, gesture recognition, and surveillance [43, 46, 47, 68]. Most of the current approaches in this area are based on vision or RF sensors. RGB camera is a ubiquitous vision system that cannot work in dark conditions [59]. Other vision sensors such as thermal, IR, and depth cameras have shortcomings such as low resolution [28], high power consumption [15, 61], and limited texture exposure [66]. In particular, the intensity maps [39, 42] collected by ToF cameras are essentially the IR maps collected by IR cameras. RF-based sensing technologies such as mmWave radar and Wi-Fi are not interfered with by visible light. Unfortunately, their sensing results are highly sparse [46, 58, 72]. Compared with these existing solutions, our system can produce high-resolution images in the dark by extracting detailed textures from off-the-shelf ToF cameras.

Image Enhancement in Low-light Conditions. There have been extensive efforts to improve the RGB image quality in low-light conditions. A multi-task learning framework is proposed in [22] to explore the intrinsic pattern behind illumination translation for object detection in poor light environments. Moreover, in [35],
thermal images are synthesized from RGB images with a Generative Adversarial Network to enable monitoring in low-light conditions. Recently, several studies have formulated the light enhancement of RGB images as a deep curve estimation problem, which requires the reference images as the input [29]. These approaches can not work in fully dark conditions without any illumination. Moreover, designed for RGB cameras, they can not be directly applied to ToF cameras due to the fundamental differences between the two sensor modalities.

ToF Augmentation. A family of techniques has been proposed for depth image enhancement, most focused on improving the noise models of ToF cameras for accurate distance measurement [40, 55, 70]. An energy-efficient epipolar imaging approach is proposed in [14] to improve the robustness of depth measurement in extreme scenarios, and the centimeter-wave and interferometric imaging are utilized in [16] to enhance the precision of iToF cameras. A recent work [66] illustrates the feasibility of extracting rich textures from depth maps. However, it requires additional hardware, such as an external IR emitter and distorts depth measurements during texture exposure, making it incompatible with current depth-based applications. In contrast, Mozart generates high-resolution texture maps based entirely on-device data processing. As a result, it not only can be implemented on mainstream off-the-shelf ToF cameras and ToF-enabled smartphones, but also can obtain high-resolution texture maps and depth maps simultaneously.

3 BACKGROUND AND MOTIVATION

In this section, we introduce the basic principles of ToF sensing, study the impact of phase component manipulation, and compare the manipulated maps with IR maps to motivate our design.

3.1 Principles of ToF Depth Sensing

Depth measurement from time-of-flight. As shown in Figure 2a, a ToF depth camera emits IR light, illuminates the scene to be captured, and receives the IR light reflected by the objects in the scene. The distance is measured based on the fact that the round trip time-of-flight \( t \) of the IR signal between the scene and the camera is strictly proportional to the distance. Specifically, we have

\[
 t = \frac{2d}{c}, \quad \text{where } d \text{ is the distance of the scene and } c \text{ is the speed of light.}
\]

Off-the-shelf ToF cameras fall into two categories based on how the time-of-flight is measured: direct Time-of-Flight (dToF) and indirect Time-of-Flight (iToF). Compared with dToF, iToF is more suitable for 3D imaging applications due to its low cost and high-resolution [66]. Most of the ToF modules on mobile devices (especially Android smartphones) in the current market adopt the iToF technology [21, 23]. Moreover, the iToF camera is expected to account for the major share of the global ToF market in the next decade [11]. Mozart is designed to work with iToF cameras, and all ToF cameras in this paper refer to iToF cameras unless otherwise indicated.

Measuring ToF based on received signal phase. The iToF camera has two successive windows (in-phase and quadrature) to receive the reflected light and uses the phase shift of the returned light to calculate the time of flight. Figure 2b illustrates the general principle of calculating the phase shift in a ToF camera. The time of flight \( t \) can be calculated by:

\[
 t = \frac{N_2}{N_1 + N_2} \cdot T_p,
\]

where \( T_p \) is the width of the pulse, \( N_1, N_2 \) are the amount of received light in successive in-phase and quadrature windows, which we refer to as phase components.

Besides the basic designs, mainstream off-the-shelf iToF cameras also adopt several advanced techniques to mitigate the influence of ambient light on distance measurement. For example, the continuous-wave iToF cameras take multiple samples per measurement (i.e., using more than two windows) to calculate the phase shift, which can reduce the energy offset caused by ambient light during the process of each distance measurement [27]. For different types of iToF cameras, we can always obtain the equivalent \( N_1, N_2 \) from their raw phase components.

3.2 A Motivation Study

Impact of Phase Components. As shown in Eqn. (1), the depth measurements are calculated from the phase components \( (N_1, N_2) \). Moreover, calculating the depth of a point in the scene is equivalent to the dimension reduction from 2-D \( (N_1, N_2) \) to 1-D distance \( d \), which inevitably loses other information like textures. In other words, the phase components contain more information about the captured scene than the depth measurement. This key observation provides opportunities for exposing detailed texture in the calculated map.

To validate this observation, we compare the depth measurements and phase components for the points across a vertical line of a human face in Figure 3a, where the blue curve denotes the normalized depth values, and the orange and green curve denote \( N_1 \) and \( N_2 \) components, respectively. We observe that the depth values are less volatile, while the phase components \( (N_1, N_2) \) fluctuate drastically. Moreover, for two points \( A \) and \( B \) with the same distance from the ToF camera, they have totally different phase components \( (N_1, N_2) \). Therefore, by exploiting such information encoded in the phase components, it is possible to show detailed texture information of different points in the scene.

We then conduct a simple manipulation operation on the phase components to show the feasibility of exposing more texture information about the scene. Specifically, we add a slight shift to \( N_1 \) when calculating depth using Eqn (1). Figure 3b shows the resulting depth...
map, which exhibits significantly finer-grained textures because shifting \( N_1 \) is equivalent to physically adding a well-modulated interfering signal [66]. Next, we apply a commonly used object detection model [13] to the original depth maps and the new maps with simple phase component manipulation. The results show that the human detection rate increases from less than 20% to more than 90%. This result clearly shows the great potential of exposing high-resolution textures from ToF cameras using phase manipulation. Moreover, the generated maps can significantly improve the performance of perception tasks, especially in the dark.

**Limitations of Existing IR-based Methods.** IR-based techniques are mainstream solutions for providing detailed textures and sensing in the dark. Most iToF cameras provide intensity maps [39, 42], which are essentially the IR maps collected by IR cameras. However, IR/intensity maps represent the amplitude of received IR signals and cannot fully expose texture information because other factors, like distance and scene structure, also affect the received signals. As Figure 4 shows, IR maps collected by IR and ToF cameras have the same key drawbacks. When an object is close-to/far from the IR/ToF camera, its texture details will be overwhelmed by saturation/lost due to the extremely weak signal strength. In particular, adding IR power to sense distant objects is infeasible on battery-sensitive mobile platforms, such as smartphones.

We then examine the face detection rate on the IR images collected by ToF cameras and compare them with the maps generated by phase component manipulation. It turns out the detection rate of IR images is merely 2% while the rate of manipulated maps is more than 80%, which indicates that the performance of IR images is significantly limited by distance. In contrast, the manipulated maps suffer less from distance. Moreover, the IR maps collected by the IR and ToF cameras show the same properties. Therefore, unless otherwise indicated, we will not differentiate IR maps collected by IR camera and ToF camera in the rest of this paper.

**Summary.** We now summarize the key observations on phase component manipulation during ToF measurement. First, the original depth maps are calculated using the phase components of the received IR signal. However, the transformation from phase components to depth suffers dimension reduction. In other words, the phase components contain more information than the depth map. Second, phase component manipulation can exploit such information and therefore expose more textures, which can be used to augment the performance of various depth applications in the dark. Moreover, phase component manipulation can overcome the key shortcomings of traditional IR images, including short sensing range due to rapid signal decay, significant noises, and the over-exposure effect.

### 4 APPLICATION SCENARIOS

**Mozart** exploits the phase components of infrared light during ToF measurements to expose high-resolution textures of the scene. As current ToF modules adopt various measures to eliminate the interference from ambient light, **Mozart** can work in all light conditions. Nevertheless, we focus on sensing in the low-light and dark environments in this work since there currently does not exist a ubiquitous high-resolution vision technology in these challenging conditions, i.e., the counterpart of RGB cameras in good lighting environments. The robust high-resolution sensing in the dark has many applications, e.g., longitudinal assessment of physical and mental health [20, 30, 53] of elders or babies. In general, **Mozart** can enable applications mainly in the following two manners.

**Mozart-only applications.** Thanks to its high-quality textures, the **Mozart** map alone can enable or augment various applications in the dark. For example, ToF-based face recognition would typically fail when the user’s face is away from the ToF camera more than 0.8 m due to the excessive noise of depth measurement. In such cases, **Mozart** maps can be applied to augment face recognition, which is an essential function for smartphones, smart door locks, and smart surveillance systems [31, 32].

Besides, **Mozart** on mobile phones can enable accurate facial expression recognition under all lighting conditions to monitor the user’s emotional state, which enables a more natural user interface adaptive to the user’s emotions [51]. Other representative applications in the dark include complex gesture recognition [43], security surveillance [68], robot navigation [46], etc. For instance, in a smart building embedded with depth ToF cameras on the wall, users can use gestures to control lights and other appliances, even in low-light and dark conditions.

**Integration with depth map.** The result of **Mozart** can not only provide high-quality input for downstream applications but also enable the integration of depth maps and rich-in-texture **Mozart** maps for new 3D applications. First, **Mozart** maps can provide a new
mechanism for training machine learning models for depth data in ToF-only systems. Specifically, the high-quality texture details can generate accurate labels by directly leveraging CV algorithms, which can be used for quick model training without manual labeling. Second, the accuracy of perception tasks can be improved by fusing the features of Mozart maps and depth maps. Finally, better 3D structures of objects [73] can be captured by combining detailed textures of Mozart maps and corresponding depth maps for ToF-only modules.

5 SYSTEM ARCHITECTURE

Mozart features a novel approach called phase component manipulation, which exploits effective mapping of the phase components during ToF measurements (i.e., \( N_1, N_2 \) in Eqn. (1)) to generate the high-resolution texture of the scene. Figure 5 shows the system architecture of Mozart. Unlike other depth camera systems that obtain depth maps directly, Mozart takes advantage of the phase components \((N_1, N_2)\) during ToF measurements. As the theoretical foundation of Mozart design, we first present an in-depth analysis of the relationship between phase components and exposed textures of the scene based on the physical reflection model for the received IR light. Then we further introduce two techniques to further enhance the exposed texture information, i.e., compensation for illumination attenuation and redistribution of total reflection outliers. With the help of the physics texture model, highly compute-efficient phase manipulation functions for exposing and enhancing textures are proposed in Section 6.2. We provide typical examples of lightweight manipulation functions and guidance in selecting effective functions for different applications. Lastly, we propose an end-to-end autoencoder-based texture generation implementation by designing highly effective learning objectives according to the physics models in Section 6.3, which automatically learns efficient representations from \( N_1, N_2 \) maps. Even though both implementations are based on our physics model, the lightweight approach is the best choice when computing resources are limited, and the autoencoder-based method performs the best in dynamic and complex scenes.

6 METHODOLOGY

This section illustrates how to expose and enhance detailed textures from ToF phase components. To this end, we first propose a first-principle physics model in Section 8.3, which provides the theoretical foundation for our texture exposure approaches in the ToF system. Then we introduce two specific techniques to further enhance the exposed texture information, i.e., compensation for illumination attenuation and redistribution of total reflection outliers. With the help of the physics texture model, highly compute-efficient phase manipulation functions for exposing and enhancing textures are proposed in Section 6.2. We provide typical examples of lightweight manipulation functions and guidance in selecting effective functions for different applications. Lastly, we propose an end-to-end autoencoder-based texture generation implementation by designing highly effective learning objectives according to the physics models in Section 6.3, which automatically learns efficient representations from \( N_1, N_2 \) maps. Even though both implementations are based on our physics model, the lightweight approach is the best choice when computing resources are limited, and the autoencoder-based method performs the best in dynamic and complex scenes.

6.1 Physics Model for Exposing Textures

In this section, we propose a physics model and leverage it to expose and enhance detailed textures from ToF phase components.

6.1.1 Modeling Textures Using Phase Components. As introduced in Section 3.1, the phase components of ToF measurements may vary for the points at the same distance due to objects’ texture, thereby essentially encoding detailed texture information besides
we know, the IR light emitted by the ToF camera will be **diffusely reflected** by the surface of objects in most cases. Therefore, we leverage the Lambertian reflection model (Figure 6a) [56] to model the process of reflection, in which the amount of received IR light reflected by the object at a distance \(d\) can be calculated by:

\[
E_d = \frac{E_0 \alpha \cos \theta}{8d^2} = \frac{E_0 \alpha}{8d^2},
\]

where \(E_0\) is a constant determined by the emission power of the ToF camera, \(\alpha\) is the reflectivity of the object and \(\theta\) is the angle of incidence. We can see that the intensity of received light is determined by objects’ reflectivity \(\alpha\), the incidence angle \(\theta\), and the distance \(d\). The former two factors together form the textures of objects [69]. In this paper, we define a new variable, "albedo" \(\beta = \alpha \cos \theta\), to quantify the two factors on the object side that have an impact on the intensity of the received light.

Based on Eqn. (2) and the physical meaning of \(N_1, N_2\) (see Section 3.1), we can establish the relationship between the phase components and the texture information \(\beta\):

\[
N_1 = \frac{E_0}{8D} \frac{\beta(D-d)}{d^2}, \quad N_2 = \frac{E_0}{8D} \frac{\beta}{d},
\]

where \(D\) is the ToF camera’s range of measurement. Note that \(E_0, D\) are all constants, thus \(N_1, N_2\) are functions of albedo \(\beta\) and distance \(d\), namely \(N_1 = n_1(\beta, d), N_2 = n_2(\beta, d)\). Therefore, the goal of phase (i.e., \(N_1, N_2\)) manipulation in Mozart is to find an optimal way of combining albedo \(\beta\) and distance \(d\) to augment texture information as much as possible for applications in the dark.

### 6.1.2 Exposing Textures: Albedo-monotonic

Given the above physics model, we show how to manipulate the phase components \(N_1, N_2\) to expose textures effectively. We formulate the phase component manipulation problem as a mapping from a 2-D vector to a scalar as:

\[
f(N_1, N_2) \rightarrow S_i,
\]

where \(i\) is the index of a pixel in the map, and \(S_i\) is the corresponding scalar in the resulting map. The function \(f(\cdot)\) will be applied to every pixel in the whole map.

The original depth maps do not contain detailed texture information because points with the same distance \(d\) but different albedos \(\beta\) can not be differentiated. Therefore, to expose detailed texture information, the phase components mapping \(f(\cdot)\) should keep the same order as albedo \(\beta\), which means that the pixel with larger \(\beta\) will always have a larger value after the mapping. In this way, for any two points A and B with the same distance \(d\) in the scene, if \(\beta_A < \beta_B\), the mapping should have \(f(N_1^A, N_2^A) < f(N_1^B, N_2^B)\). Therefore we have \(\frac{\partial f(N_1, N_2)}{\partial N_1} > 0\). Combining with Eqn. (3), we have the following constraints of the phase manipulation mapping:

\[
\frac{\partial f(N_1, N_2)}{\partial N_1}(D-d) + \frac{\partial f(N_1, N_2)}{\partial N_2}d > 0
\]

Eqn. (5) ensure that the transformed result \(f(N_1, N_2)\) is monotonically increasing in terms of albedo \(\beta\) at a given distance \(d\), which we refer to as **albedo-monotonic**. Such monotonicity keeps the same structure in the transformed map as the albedo map, thus exposing detailed textures without introducing any artifacts (see Figure 6b). It is worth mentioning that if the inequalities in Eqn. (5) are completely opposite to the current ones, the monotonicity will still hold. However, the texture structure in the transformed map will be reversed to albedo, resulting in “negative images” [34]. For example, negative images are useful for enhancing white or grey detail embedded in dark regions of an image. Moreover, note that Eqn. (5) is a sufficient but not necessary condition, which means all transformations that meet this constraint can effectively expose textures.

#### 6.1.3 Enhancing Textures: Illumination Compensation

According to Eqn. (3), the phase components \(N_1 = n_1(\beta, d)\) and \(N_2 = n_2(\beta, d)\) decrease with the distance. Therefore, as shown in Figure 7, in a typical map generated by phase manipulation, the near objects will be much brighter than distant objects, reducing the utility of distant points of the image. More specifically, the average intensity of received light decreases drastically with the distance. Therefore, in this section, we propose to compensate for the illumination attenuation introduced by longer distances to further enhance the exposed textures during phase manipulation. To achieve this, we
we should expand the dense value around 0 and compress the sparse distributions at larger values. Then the mapping results of these outliers through the non-decreasing functions \( f(S) \), where the dense distribution can be squeezed. Therefore, \( r(0) \) can be expanded, and the sparse distribution at larger values can be squeezed. Therefore, \( r(S) \) can limit the bound of higher outlier values. The lower part of Figure 8 shows the texture map before and after the redistribution of phase manipulation, where the manipulated map after redistribution has a substantially higher contrast and more uniform distribution of pixel values.

**6.2 Light-weight Phase Manipulation**

Given the physics model and principles of exposing and enhancing textures proposed in Section 6.1, we introduce how to design efficient functions \( f(N_1, N_2) \) in practice to expose textures.

**Selecting functions for texture exposure.** The first stage of designing efficient phase manipulation functions is to check whether the candidate functions are albedo-monotonic. In Figure 9, we show the generated maps by applying the following functions: 

\[
\begin{align*}
\text{(a)} \quad f_0 &= N_2 / N_1, \\
\text{(b)} \quad f_1 &= N_1 N_2 / (N_1 + N_2), \\
\text{(c)} \quad f_3 &= N_1 N_2^2.
\end{align*}
\]

Figure 9: The functions with large polynomial degrees will turn normal values into outliers, resulting in over-exposure.

To alleviate the impact of outliers introduced by total reflections, we should expand the dense value around 0 and compress the sparse outliers with large values. To achieve this, we add a new mapping function \( r(S) \) outside the function \( f(\cdot) \) defined in Eqn. (5), where \( S \) denotes the transformed result of \( f(N_1, N_2) \). Here the continuous function \( r(S) \) must be monotonically increasing, and the rate of increase gets smaller with the pixel value \( S \). Figure 8 shows an example of the redistribution function \( r(S) \), where the dense distribution at 0 can be expanded, and the sparse distribution at larger values can be squeezed. Therefore, \( r(S) \) can limit the bound of higher outlier values. The lower part of Figure 8 shows the texture map before and after the redistribution of phase manipulation, where the manipulated map after redistribution has a substantially higher contrast and more uniform distribution of pixel values.

**6.1.4 Enhancing Textures: Redistribution of Outliers.** Based on the physics model proposed in Section 6.1.2, we are able to expose texture information through ToF phase manipulation. In practice, there will always exist total reflection on the surface of certain objects, such as metals and glass. Therefore, the points with total reflection will not satisfy the Lambertian reflection model in Eqn. (2), and the corresponding \( N_1, N_2 \) received by the ToF camera will far exceed the normal values. For example, a typical value of \( N_2 \) is less than 100, while the \( N_2 \) value of total reflection can be more than 1,000. Then the mapping results of these outliers through the non-decreasing functions defined in Eqn. (5) will also exceed the normal range. As shown in Figure 8, if we normalize the texture map to grayscale, most of the pixels will be squeezed into a small range near 0, while the points with total reflection exhibit isolated bright spots. Therefore, in this section, we introduce how to enhance the exposed textures during phase manipulation by redistributing the total reflection outliers.

To alleviate the impact of outliers introduced by total reflections, we should expand the dense value around 0 and compress the sparse outliers with large values. To achieve this, we add a new mapping function \( r(S) \) outside the function \( f(\cdot) \) defined in Eqn. (5), where \( S \) denotes the transformed result of \( f(N_1, N_2) \). Here the continuous function \( r(S) \) must be monotonically increasing, and the rate of increase gets smaller with the pixel value \( S \). Figure 8 shows an example of the redistribution function \( r(S) \), where the dense distribution at 0 can be expanded, and the sparse distribution at larger values can be squeezed. Therefore, \( r(S) \) can limit the bound of higher outlier values. The lower part of Figure 8 shows the texture map before and after the redistribution of phase manipulation, where the manipulated map after redistribution has a substantially higher contrast and more uniform distribution of pixel values.

**6.2 Light-weight Phase Manipulation**

Given the physics model and principles of exposing and enhancing textures proposed in Section 6.1, we introduce how to design efficient functions \( f(N_1, N_2) \) in practice to expose textures.

**Selecting functions for texture exposure.** The first stage of designing efficient phase manipulation functions is to check whether the candidate functions are albedo-monotonic. In Figure 9, we show the generated maps by applying the following functions: 

\[
\begin{align*}
\text{(a)} \quad f_0 &= N_2 / N_1, \\
\text{(b)} \quad f_1 &= N_1 N_2 / (N_1 + N_2), \\
\text{(c)} \quad f_3 &= N_1 N_2^2.
\end{align*}
\]

Figure 9: The functions with large polynomial degrees will turn normal values into outliers, resulting in over-exposure.

To alleviate the impact of outliers introduced by total reflections, we should expand the dense value around 0 and compress the sparse outliers with large values. To achieve this, we add a new mapping function \( r(S) \) outside the function \( f(\cdot) \) defined in Eqn. (5), where \( S \) denotes the transformed result of \( f(N_1, N_2) \). Here the continuous function \( r(S) \) must be monotonically increasing, and the rate of increase gets smaller with the pixel value \( S \). Figure 8 shows an example of the redistribution function \( r(S) \), where the dense distribution at 0 can be expanded, and the sparse distribution at larger values can be squeezed. Therefore, \( r(S) \) can limit the bound of higher outlier values. The lower part of Figure 8 shows the texture map before and after the redistribution of phase manipulation, where the manipulated map after redistribution has a substantially higher contrast and more uniform distribution of pixel values.
like \( r_3(S) \) to better redistribute the total reflection outliers. On the contrary, we can choose a more mild function like \( r_1(S) \) so that the distribution will not affect the pixels with typical values.

6.3 Autoencoder-based Phase Manipulation
The function-based texture generation in Section 6.2 requires careful design and manual tuning for different applications, which is labor-intensive and requires substantial domain expertise. Therefore, we propose an end-to-end autoencoder-based texture generation approach, which automatically learns efficient representations from \( N_1, N_2 \) maps. Our key idea is to utilize the physics models for texture exposure and enhancement proposed in Section 6.1 to design highly effective learning objectives for the autoencoder.

Our approach has several key advantages. First, the autoencoder neural network is trained in an unsupervised manner, which does not require manual labeling or reference images, in contrast to previous supervised image enhancement solutions [48]. Second, by exploiting the physics models for texture exposure and enhancement proposed in Section 6.1, we design several effective loss functions to train the deep autoencoder neural network. As a result, the autoencoder can output high-resolution texture maps without introducing artifacts or large noises. Third, due to the convolutional layers, autoencoder-based methods can capture local spatial information within the receptive field of convolutional kernels. Finally, compared with the manually crafted manipulation functions, the autoencoder network is more scalable in generating high-resolution \( \text{Mozart} \) maps for different applications. For example, for the two applications (e.g., human tracking and gesture recognition) with different outlier distributions or IR illumination attenuation effects, the autoencoder-based texture generation can be directly applied without any modification or manual system tuning.

6.3.1 Autoencoder Design. Autoencoder is a widely-used unsupervised learning approach in computer vision tasks that can learn efficient features from unlabeled data [37, 50]. We use a deep autoencoder neural network as a generative model to adaptively generate high-resolution \( \text{Mozart} \) maps from \( N_1, N_2 \) maps. As shown in Figure 10, the autoencoder neural network has two main components: the encoder and the decoder network. During the training of the deep autoencoder, the phase component \( N_1, N_2 \) maps will be input to the neural network. Then the encoder network maps the input \( N_1, N_2 \) maps into deep latent space, and the decoder network reconstructs high-dimension \( \text{Mozart} \) maps from the deep embeddings. In this way, the autoencoder neural network can learn invariant features underlying the \( N_1, N_2 \) maps collected from different scenarios [44]. We use a 3D-CNN for the encoder and decoder to explore the inter-channel relationships between \( N_1, N_2 \) maps. Finally, the output maps will be used to calculate the unsupervised training loss, where the loss functions are designed based on the physics models for texture exposure and enhancement proposed in Section 6.1.

The goal of training the autoencoder neural network is to learn efficient representations from \( N_1, N_2 \) maps to generate high-resolution texture maps. Therefore, similar to the lightweight mapping function in Section 6.2, the autoencoder is trained to exploit efficient manipulations to the phase components for exposing texture information.

6.3.2 Design of Loss Functions. To train an autoencoder neural network that can generate high-resolution texture maps, we carefully design three loss functions according to the physics models proposed in Section 6.1, including the albedo similarity loss, the illumination attenuation loss, and the uniform distribution loss. Suppose \( P \) denotes the \( \text{Mozart} \) output of the autoencoder neural network. We design the three loss functions as follows.

**Albedo similarity loss.** Unlike the lightweight phase manipulation functions that can maintain the pixel topology of \( N_1, N_2 \) maps, the neural network-based method is more like a black box, which may generate artificial textures that do not exist in the actual scene. As shown in Section 6.1, the albedo map calculated by the manipulation function \( f(\cdot) \) contains textures of the scene. Therefore, we use the structural similarity [64] between the \( \text{Mozart} \) output and the corresponding albedo map to guide the training of the \( \text{Mozart} \) Autoencoder model. Suppose \( B \) denotes the albedo map, then the albedo similarity loss is calculated as follows:

\[
\mathcal{L}_S = 1 - S(P, B),
\]

where \( S(X, Y) \in [0, 1] \) denote the structural similarity index of two images \( X \) and \( Y \). A larger \( S(X, Y) \) means more similarity between the map \( X \) and \( Y \).

**Illumination compensation loss.** As shown in Section 6.1.3, the phase components \( N_1 \) and \( N_2 \) decrease with the distance, making the near objects much brighter than distant objects. Therefore, we propose an illumination compensation loss to penalize the high-intensity pixels near the ToF camera.

By design, the depth maps have larger distance values for distant objects and smaller distance values for close objects. Therefore, the depth maps (or maps positively correlated to distance) can serve as a reference kernel to correct the uneven light field of \( \text{Mozart} \) output. Moreover, as the depth maps usually have lots of noises that can affect the quality of generated \( \text{Mozart} \) maps, we use the denoised depth maps \( D \) after median filter [19] to calculate the light compensation loss:

\[
\mathcal{L}_L = \| P \circ D \|_1,
\]

where \( \circ \) denotes the Hadamard product.
Three ToF modules.

for a map.

\( c \)

where we change the generated map to grayscale with the value

\( \lambda \)

where we change the generated map to grayscale with the value range [0, 255] and \( p_c(P) \) is the normalized histogram counts of value \( c \) for a map.

Overall Training Loss Function. Putting the above three loss functions together, the overall loss function for training the autoencoder neural network is:

\[
    \mathcal{L} = \lambda_3 \mathcal{L}_S + \lambda_1 \mathcal{L}_L + \lambda_u \mathcal{L}_U
\]

Here \( \lambda_3, \lambda_1, \lambda_u \) are the coefficients that weigh the contribution of each loss function and can be adjusted easily in different applications. For example, when the depth maps are very noisy, we can set a smaller \( \lambda_1 \) to reduce the impact of depth on the light compensation, leading to smaller noise in the generated \( \text{Mozart} \) maps.

Table 2: Summary of the mobile phones with \textit{Mozart} implemented. The resolution refers to the typical resolution obtained through the corresponding API (instead of the physical resolution of the ToF sensor on the mobile phone). All resolutions in the table are sufficient for typical applications such as faceID at reasonable distances.

<table>
<thead>
<tr>
<th>Phone Model</th>
<th>ToF Camera(s)</th>
<th>ToF API</th>
<th>Resolution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Huawei P30 Pro</td>
<td>rear</td>
<td>AREngine</td>
<td>240 × 180</td>
</tr>
<tr>
<td>Huawei Mate30 Pro</td>
<td>rear &amp; front</td>
<td>AREngine</td>
<td>240 × 180</td>
</tr>
<tr>
<td>Samsung S20 Ultra</td>
<td>rear</td>
<td>ARCore</td>
<td>640 × 480</td>
</tr>
</tbody>
</table>

Uniform distribution loss. As shown in Section 6.1, the outlier values introduced by total reflection will distort the distribution uniformly. However, when the histogram of a map is uniform across the entire value range, the map will have a higher contrast [57]. This feature is also friendly to many existing object detection and tracking algorithms [41, 45]. Therefore, consistent with the redistribution function in Section 6.1.4, we design a uniform distribution loss by minimizing the negative histogram entropy for the output map:

\[
    \mathcal{L}_U = \frac{1}{255} \sum_{c=0}^{255} p_c(P) \log p_c(P),
\]

where we calculate the phase components from ToF cameras are not available (i.e., the confidence maps in Android documentation), which can be obtained from all Android smartphones using \textit{Camera2 API} [4]. Moreover, smartphones from a few manufacturers provide dedicated ToF APIs to access the ToF data. For example, \textit{AREngine} [3] available on Huawei devices can provide 3-bit confidence maps and 13-bit depth maps. \textit{ARCore} [2] available on Google-certified models such as Samsung S20 Ultra can provide 8-bit confidence maps and 16-bit depth maps. After obtaining the phase components, we calculate the \textit{Mozart} maps using manually designed functions on the phones for downstream applications. \textit{Mozart} on smartphones is implemented using Java on Android Studio. Note that the latency of generating a single \textit{Mozart} map on all three smartphones is merely around 15 ms (see Figure 12a), thus enabling real-time applications.

Edge Platforms. We also implement \textit{Mozart} with three mainstream standalone ToF cameras (Figure 11b), including DMOM2508 [26], Vzense DCAM710 [12], and DepthEye Wide ToF [5], on Nvidia Jetson Xavier [7]. The DepthEye ToF camera adopts an IMX556PLR CMOS from Sony, from which we can easily obtain the phase components of ToF measurements directly via APIs. The phase components of the other two ToF cameras need to be calculated indirectly from depth maps and intensity maps. \textit{Mozart} on Nvidia Xavier (running Ubuntu 18.04) is implemented using C++ and Python.

On edge platforms, we implement two different texture generation approaches, including \textit{Mozart-manual}, where we manually select the best functions for different applications according to the principles proposed in Section 6.2, and \textit{Mozart}, where we train the autoencoder neural network using the loss functions defined in Section 6.3. We note that \textit{Mozart-manual} requires substantial efforts and domain expertise to choose a proper function in a trial-and-error manner. Moreover, such a labor-intensive process must be repeated for different applications. On the contrary, the proposed autoencoder can automatically learn efficient representations from phase components to generate texture maps while achieving similar performance with \textit{Mozart-manual}.

Figure 11: Implementation of \textit{Mozart} with smartphones and standalone ToF cameras.

7 SYSTEM IMPLEMENTATION AND OVERHEAD

7.1 Implementation on Various Platforms

Smartphones Platforms. A number of smartphones (e.g., Huawei P/Mate series, Samsung S/Note series) are equipped with ToF cameras for various applications such as FaceID, In-Air Gesturing, and AR/VR. We first implement \textit{Mozart} on three off-the-shelf smartphones with embedded ToF cameras, whose specifications are shown in Table 2. To demonstrate the real-time performance of Mozart, we built an Android App (Figure 11a) that can help users identify objects under dark environments. The demo video (https://youtu.be/qBEffXVft_8) shows that, compared with depth maps, the Mozart App can provide substantially more texture details in the dark environment in real time.

The raw phase components from ToF cameras are not available on Android smartphones. To address this challenge, we calculate the phase components indirectly from depth and intensity maps (i.e., the confidence maps in Android documentation), which can be obtained from all Android smartphones using \textit{Camera2 API} [4]. Moreover, smartphones from a few manufacturers provide dedicated ToF APIs to access the ToF data. For example, \textit{AREngine} [3] available on Huawei devices can provide 3-bit confidence maps and 13-bit depth maps. \textit{ARCore} [2] available on Google-certified models such as Samsung S20 Ultra can provide 8-bit confidence maps and 16-bit depth maps. After obtaining the phase components, we calculate the \textit{Mozart} maps using manually designed functions on the phones for downstream applications. \textit{Mozart} on smartphones is implemented using Java on Android Studio. Note that the latency of generating a single \textit{Mozart} map on all three smartphones is merely around 15 ms (see Figure 12a), thus enabling real-time applications.

The raw phase components from ToF cameras are not available on Android smartphones. To address this challenge, we calculate the phase components indirectly from depth and intensity maps (i.e., the confidence maps in Android documentation), which can be obtained from all Android smartphones using \textit{Camera2 API} [4]. Moreover, smartphones from a few manufacturers provide dedicated ToF APIs to access the ToF data. For example, \textit{AREngine} [3] available on Huawei devices can provide 3-bit confidence maps and 13-bit depth maps. \textit{ARCore} [2] available on Google-certified models such as Samsung S20 Ultra can provide 8-bit confidence maps and 16-bit depth maps. After obtaining the phase components, we calculate the \textit{Mozart} maps using manually designed functions on the phones for downstream applications. \textit{Mozart} on smartphones is implemented using Java on Android Studio. Note that the latency of generating a single \textit{Mozart} map on all three smartphones is merely around 15 ms (see Figure 12a), thus enabling real-time applications.

On edge platforms, we implement two different texture generation approaches, including \textit{Mozart-manual}, where we manually select the best functions for different applications according to the principles proposed in Section 6.2, and \textit{Mozart}, where we train the autoencoder neural network using the loss functions defined in Section 6.3. We note that \textit{Mozart-manual} requires substantial efforts and domain expertise to choose a proper function in a trial-and-error manner. Moreover, such a labor-intensive process must be repeated for different applications. On the contrary, the proposed autoencoder can automatically learn efficient representations from phase components to generate texture maps while achieving similar performance with \textit{Mozart-manual}.

7.2 System Overhead

In this section, we evaluate the system overhead of \textit{Mozart} on three smartphones and Nvidia Jetson Xavier with three mainstream standalone ToF modules. First, we developed a smartphone App to classify facial expressions from a typical expression set [60] (i.e., angry, disgust, fear, happy, sad, surprise, and neutral) under...
different illumination conditions. Results show that Mozart on smartphones outperforms depth maps by 20% and IR maps by 40% in mean recognition accuracy. We further implemented an object detection task on Nvidia Xavier with the threeToF modules. The results indicate that the performance improvement of Mozart maps is significant for allToF cameras. Specifically, on the Vseenze TOF camera, Mozart outperforms depth and IR maps by 63.76% and 45.28%, respectively.

System Overhead on Smartphones. We compare the system overhead of the native TOF system, Mozart-manual, and inference with Mozart maps in terms of latency, power consumption, and memory usage. We use PerfDog [8] to measure the overall power consumption and memory usage of each task on three smartphones. The results are shown in Figure 12a. First, the latency of calculating a single Mozart map on mobile phones is smaller than 14.5 ms, which can easily support real-time applications with a frame rate of 30 fps. Moreover, calculating Mozart maps does not significantly increase any resource consumption, including power consumption and memory usage. It is worth noting that the average latency for each major step (i.e., sensor sampling, phase components calculation, and Mozart map generation) is 0.06 ms, 10.92 ms, and 7.13 ms, respectively. The fact implies that the latency can be significantly improved if Mozart can access native phase components through APIs since there is no phase components calculation. Moreover, the image quality of Mozart can also be improved thanks to more accurate native phase components.

System Overhead on Edge Devices. We compare Mozart against Mozart-manual and the native depth system that generates depth maps from TOF phase components. As discussed earlier, Mozart-manual is designed by an extensive search of compute-efficient texture generation functions, which requires substantial efforts and domain expertise. As a result, the computing overhead of Mozart-manual at runtime is expected to be smaller than autoencoder-based Mozart. During the end-to-end experiments, we measure the averaged computation time and power consumption for obtaining one map frame. The power consumption is obtained using tegracostats [9] provided by Nvidia.

The results are shown in Figure 12b. First, Mozart-manual outperforms the native depth system in computation time and power consumption. This shows that the phase manipulation functions we designed for Mozart-manual are more efficient than the built-in transformation of phase components to depth maps. Specifically, Mozart-manual produces each frame in merely 43.5ms, i.e., at a rate of about 23 fps, which allows it to be executed in real time on embedded platforms. The autoencoder-based Mozart takes more time and power consumption, while it can still achieve about 12 fps, which is acceptable in most depth applications [33, 66]. We note that the system overhead can be further reduced by various existing techniques [49], including optimizing the computation pipeline in a hardware-software co-optimization and adopting a sparse autoencoder, which is left for future work.

8 COMPARISON WITH OTHER SENSOR MODALITIES

In this section, we compare Mozart with the baseline systems that employ the following sensor modalities: RGB, RGB-enhanced, Depth, IR and mmWave Radar, using three new real-world datasets collected in dark environments. The RGB-enhanced maps are generated by Zero-DCE [29], a state-of-the-art image enhancement approach from computer vision literature.

8.1 Data Collection in the Dark

The reasons for collecting the new datasets are as follows. First, existing depth camera-based datasets do not contain corresponding samples of other sensor modalities. Besides, there are few multimodal datasets (including RGB/Depth/IR/Radar) collected under dark environments, which is the main application scenario of Mozart. The three datasets consist of over 1,000,000 frames with a total of 33 subjects.

For a fair comparison, we collect the data of all these sensors and the TOF phase components simultaneously at 10 Hz. The RGB images are collected using the Vzense camera [12], and the IR/depth images are collected using the DepthEye TOF camera [5]. The radar point clouds are collected using a 60-64GHz mmWave radar TI IWR6843 [10]. We convert the IR, RGB, RGB-enhanced, Depth, and Mozart maps to the same dimension (640, 480) and apply the same detection or classification models in each task. We convert each frame of radar points into voxels of a fixed dimension (2, 16, 32, 16) and apply existing detection or classification models to them for each task. We convert each frame of radar points into voxels of a fixed dimension (2, 16, 32, 16) and apply existing detection or classification models in each task.

Human tracking dataset. Continuous human tracking in the dark is a typical task in applications like security surveillance, which need to capture high-resolution textures of the scene. As shown in
We first compare Mozart against different sensing approaches for the three datasets in the dark. For the human tracking task, we use a widely-used object detector YOLOv5 [13] to detect the persons in each frame, which will output the predicted objects and the prediction confidence for each object. However, this evaluation is not applicable to radar point clouds as they do not support semantic-based tasks due to the data sparsity. Therefore, the radar baseline in this experiment only tracks moving objects in the scene without detecting the type of objects (i.e., the person). For the face recognition task, we first detect the faces in the image maps using a pre-trained RetinaFace detector [24]. Then a pre-trained ArcFace [25] model transforms the detected face areas into 512-dimension feature vectors. For a fair comparison, we directly trained a neural network on the detected face areas in depth maps instead of applying ArcFace. Moreover, the accuracy of depth maps is calculated only based on the maps with face areas detected. For voxels of radar data, we train a 3D-CNN model to classify the faces of 12 different people. To recognize the hand gestures, we first detect and localize the hands by applying the MediaPipe [6] to the RGB, depth, IR, and Mozart maps. Then we input the cropped hands area into a lightweight 2D-CNN model to classify the gestures. The radar voxels are trained using a 3D-CNN model directly to classify 20 different gestures.

The results of different datasets are shown in Figure 14. First, the baselines perform very poorly for applications in the dark. For example, the RGB and depth images only achieve 1.66% and 1.54% mean accuracy in the face recognition task. Second, both Mozart and Mozart-manual consistently outperform other baselines in different datasets with various settings. Moreover, Mozart can approach or even surpass the performance of the manually designed Mozart-manual in different datasets, which shows that the autoencoder texture generation is robust and scalable for various applications and the three loss functions are sufficient to generate high-quality Mozart maps in most cases.

8.3 Performance in Dynamic Conditions

Now we evaluate Mozart’s performance under dynamic conditions, including different environments, illumination conditions, and distances of objects. The results are shown in Figure 16.

Different Environments. We first evaluate the impact of different environments in human tracking, including the square, room, and corridor. First, both Mozart-manual and Mozart have a robust performance across different environments, consistently outperforming all baselines. Radar performs extremely poorly in rooms/corridors due to significant multi-path effects. RGB images have extremely low accuracy for all settings in the dark. Depth and IR maps perform poorly in the squares where people walk in a large range (e.g., 5-10 m).

Different Light Conditions. We then evaluate the impact of different light conditions in face recognition, where Light 1, 2, 3 have sequentially increasing ambient light intensity. First, the performance of Mozart and Mozart-manual are very stable under different light conditions and outperforms all baselines except for RGB images under Light 3 (full illumination). Besides, depth, IR, and radar also yield a stable performance under different light conditions as they do not rely on ambient light. However, the performance of RGB drops drastically with lower light levels since it is highly susceptible to non-ideal environmental light conditions.

Different Distances of Objects. Lastly, we evaluate the impact of different distances of objects in face recognition. Almost all modalities perform worse in the far setting (1.5 m) because the size of face areas is smaller than that under the near setting (0.5 m). However, both Mozart-manual and Mozart suffer subtle performance degradation and always outperform all baselines.

8.4 Ablation Study

In this section, we evaluate the effectiveness of different design components for texture generation based on the human tracking
task. Both Mozart-manual and Mozart are implemented using different component combinations. Specifically, exposing textures based on the physics model is denoted as C1, redistribution of total reflection outliers is denoted as C2, and compensation for illumination attenuation is denoted as C3.

The results are shown in Figure 15. First, the Mozart-manual with C1+C2 already shows significant accuracy improvement. Second, the results with different loss combinations of Mozart all show significant accuracy improvement (i.e., at least 20%) over IR maps. Lastly, the performance of Mozart is more stable than Mozart-manual under different configurations, which shows the robustness of autoencoder-based texture generation that exploits the physics texture models.

9 DISCUSSION
In this section, we discuss several future directions of our work.

Applying Mozart maps in multi-modality vision algorithms. In this paper, we investigate the feasibility of directly applying existing uni-modal vision algorithms to Mozart. We will study the effect of using Mozart to substitute a specific modality in multi-modal vision algorithms. Specifically, the RGB/IR fusion algorithm [38] will perform worse if we replace IR images with Mozart maps. The performance degradation comes from the discrepancy between IR images and Mozart maps features and their different correlation with RGB images. Therefore, we will study the feasibility of applying Mozart to multi-modal vision algorithms and propose practical measures to mitigate the domain gap between Mozart maps and other vision modalities. Similar issues can be studied when Mozart maps replace RGB images in RGB-D systems [18], RGB-language models [17], RGB-audio fusion [36], etc.

Impact of Mozart on future depth systems. In addition to the new applications enabled by Mozart that we discussed in the paper, we will study the impact of Mozart on future depth-sensing systems.

- Privacy/Security issues of ToF cameras. ToF cameras were previously considered privacy-preserving. However, our work has clearly illustrated the potential risk of privacy leakage in ToF cameras. Therefore, an important and urgent question is how to use the ToF system in a privacy-preserving manner. Fortunately, we can easily control how much Mozart exposes detailed textures through phase manipulation. Therefore, a possible paradigm of using ToF systems in the future is to authorize a specific degree of visual privacy exposure according to application scenarios. On the other hand, the emergence of systems like Mozart will motivate the exploration of new privacy-preserving depth measurement principles.

- The new APIs of ToF cameras. Our results show the importance of opening access to raw phase components on ToF devices. Future ToF depth systems can provide three API layers, i.e., the raw data layer providing phase components, the manipulation layer providing tool chains of phase manipulation for generating customized maps, and the result layer providing pre-defined manipulation formulas and autoencoder-based texture maps. These new APIs will enable a new generation of depth sensing in the future.

10 CONCLUSION
In this paper, we present Mozart, a new sensing system that leverages off-the-shelf ToF depth camera to generate high-resolution and rich-in-texture maps for low-light and dark scenes. Extensive experiments show that Mozart significantly outperforms existing sensing technologies and can work on smartphones and edge platforms in real time. In the future, we will study how to combine the native depth maps and Mozart maps for advanced 3D sensing applications such as 3D reconstruction of dark environments.

ACKNOWLEDGEMENT
This work is supported in part by Research Grants Council (RGC) of Hong Kong under General Research Fund #14209619, and National Natural Science Foundation of China under Grant No. 62032021.

APPENDIX
The research artifact accompanying this paper is available via https://doi.org/10.5281/zenodo.7919580.


