

**METHODS FOR PHOTOGRAPHIC  
STEGANOGRAPHY AND RADAR OBJECT SHAPE  
INFERENCE**

**By**

**ERIC WENGROWSKI**

**A dissertation submitted to the**

**School of Graduate Studies**

**Rutgers, The State University of New Jersey**

**in partial fulfillment of the requirements**

**for the degree of**

**Doctor of Philosophy**

**Graduate Program in Electrical and Computer Engineering**

**Written under the direction of**

**Kristin Dana**

**and approved by**

---

---

---

---

**New Brunswick, New Jersey**

**MAY, 2019**

## **ABSTRACT OF THE DISSERTATION**

# **Methods for Photographic Steganography and Radar Object Shape Inference**

**by Eric Wengrowski**

**Dissertation Director:**

**Kristin Dana**

In this work, we explore the fundamental problems associated with Photographic Steganography, the process of discretely sending information camouflaged in natural images from electronic display to camera. Broadly stated, the goals are minimizing the perceived visual impact of adding a new message to an image, while simultaneously maximizing the ability to accurately recover this message camera-side. This process is complicated by the photometric and radiometric effects of cameras, electronic displays, and their relative geometry and illumination conditions. In Chapter 2, we model these effects jointly as a Camera-Display Transfer Function (CDTF) and introduce two online radiometric calibration techniques to mitigate the effects of the CDTF. In Chapter 3, we extend photographic steganography by modeling and predicting color shifts that minimize perceptual impact and maximize accurate camera recovery. In Chapter 4, we use deep convolutional neural networks to jointly learn a steganographic embedding and recovery algorithm that requires no multi-frame synchronization, one of the most significant practical barriers to success for photographic steganography. The proposed techniques have all been implemented in real-time demos using consumer-grade displays



and smartphone cameras. This body of work represents a fundamental contribution to the field of camera-display communication and photographic steganography. Chapter 5 explores how computer vision techniques can be extended to monostatic radar for shape recognition.

## Acknowledgements

I would like first to thank my doctoral committee: Kristin Dana, Peter Meer, Vishal Patel, and Anthony Hoogs.

I would like to thank Peter Meer for valuable discussions on metamers in human vision. I would also like to thank Gradeigh Clark and Thomas Papathomas for our insightful discussions on human perception, and Jane Baldwin for generously lending several cameras. Finally we would like to thank my labmates Wenjia Yuan, Hang Zhang, Elie Rosen, Parneet Kaur, Thomas Shyr, Matthew Purri, Jia Xue and Blerta Lindqvist for their time and thoughtful suggestions. I would also like to thank my other collaborators Andrew Huston, Harry Sun, Viet Nguyen, Yaqin Tang, Wenjun Hu, and Ashwin Ashok for their excellent work.

I am especially grateful to the Rutgers University Electrical and Computer Engineering faculty and staff including Peter Meer (again), Athina Petropulu, Narayan Mandayam, Marco Gruteser, Hana Godrich, Saman Zonouz, Vishal Patel, Janne Lindqvist, Roy Yates, Yanyong Zhang, Grigore Burdea, Waheed Bajwa, Shantenu Jha, John Scafidi, Steve Orbine, John McCarthy, Ora Titus, Christy Lafferty, as well as Lingyi Xu, all of whom have provided me with guidance and support throughout my academic journey.

I would like to express an endless amount of gratitude and appreciation to my committee chair and Ph.D. advisor, Professor Kristin Dana, whose guidance and encouragement elevated my work to a level beyond anyone's expectations. She continues to provided me with the support for my academic and entrepreneurial career goals. She

has set an incredible example of a successful life-work balance for my labmates and I, and she has always prioritized our time over hers. I appreciate every extra mile that you went for all of us, and I hope to do the same for eager minds in the future.

Finally, I would like to thank my family, my friends Luis Garcia, Gabriel Salles-Loustau, Tim Phan, Tuan Le, Gradeigh Clark, and my lady-friend Jane Baldwin for their incredible support throughout this 6-year chapter in my career. I hope that I have graduated to become a better partner, a better friend, and a better son.

This work would not have been possible without the financial support of the Rutgers Electrical and Computer Engineering Department, the Graduate Assistance in Areas of National Need (GAANN) fellowship program from the Department of Education and Athina Petropulu, my initial support from the National Science Foundation (NSF) under grant CNS-1065463, and Lockheed Martin Corporation (LMC). I would like to thank Rowland Escritor and Kevin Vance for their substantial efforts coordinating and facilitating this funding from LMC. The Titan X used for this research was donated by the NVIDIA Corporation.

## Dedication

*This thesis is dedicated to my parents, Edward Wengrowski and Maureen Early, my first science teachers.*

## Table of Contents

<b>Abstract</b> . . . . .	ii
<b>Acknowledgements</b> . . . . .	iv
<b>Dedication</b> . . . . .	vi
<b>List of Tables</b> . . . . .	x
<b>List of Figures</b> . . . . .	xi
<b>1. Introduction</b> . . . . .	1
<b>2. Optimal Radiometric Calibration for Camera-Display Messaging</b> . .	4
1. Related Work . . . . .	8
2. System Properties . . . . .	10
3. Methods . . . . .	11
3.1. Photometry of Display-Camera systems . . . . .	11
3.2. Message Structure . . . . .	13
3.3. Optimal Online Radiometric Calibration . . . . .	15
3.4. Hidden Ratex . . . . .	18
4. Results . . . . .	21
5. Discussion . . . . .	22
<b>3. Reading Between the Pixels: Photographic Steganography for Camera- Display Communication</b> . . . . .	26
1. Introduction . . . . .	26
2. Background and Related Work . . . . .	29
3. Photographic Steganography System Design . . . . .	32

4.	Learning New Differential Metamers . . . . .	33
4.1.	Clustering Training Data . . . . .	34
5.	Experiments . . . . .	41
6.	Results . . . . .	43
7.	Discussion . . . . .	46
<b>4.</b>	<b>Light Field Messaging . . . . .</b>	<b>49</b>
1.	Introduction . . . . .	49
2.	Related Work . . . . .	51
3.	Methods . . . . .	54
3.1.	Camera-Display 1M Dataset . . . . .	56
3.2.	Training $T()$ . . . . .	59
3.3.	Training $E()$ and $R()$ . . . . .	59
4.	Experiments and Results . . . . .	60
5.	Discussion . . . . .	66
<b>5.</b>	<b>Deep CNNs as a Method to Classify Rotating Objects based on Mono- static Radar Cross Section . . . . .</b>	<b>68</b>
1.	Introduction . . . . .	68
2.	Related Work . . . . .	71
3.	Generating RCS Signals . . . . .	73
3.1.	Generalized Euler Motion . . . . .	73
3.2.	Randomizations in Motion Parameters . . . . .	74
3.3.	Update Rate, Swerling, Gaussian Noise, Gradients, and Pyramids . . . . .	75
4.	Experiments . . . . .	76
4.1.	Residual Network . . . . .	76
4.2.	Expanding the A4 Dataset . . . . .	78
4.3.	Siamese Network . . . . .	78
4.4.	Robustness Test . . . . .	81
4.5.	Refiner Network . . . . .	81

5.	Results . . . . .	83
5.1.	Classification on A4 and B4 Datasets . . . . .	83
5.2.	Classification on A5 Dataset . . . . .	88
5.3.	Robustness Metric Performance . . . . .	89
5.4.	Classification on Refined Dataset . . . . .	91
6.	Discussion . . . . .	93
<b>6.</b>	<b>Conclusion . . . . .</b>	<b>94</b>
	<b>Bibliography . . . . .</b>	<b>95</b>

## List of Tables

2.1. Main Result: Message Recovery for small $\kappa$ at $45^\circ$ oblique view . . . . .	22
2.2. Message Recovery for small $\kappa$ at $0^\circ$ oblique view . . . . .	23
2.3. Message Recovery for large $\kappa$ at $45^\circ$ oblique view . . . . .	24
2.4. Message Recovery for large $\kappa$ at $0^\circ$ oblique view . . . . .	25
3.1. Camera recovery error for various clustering methods . . . . .	42
3.2. BER for various embedding schemes . . . . .	43
3.3. Message embedding with intensity vs differential metamers example . .	45
3.4. Photographic Steganography transferred to a new camera-display pair .	48
4.1. BER for various camera-display pairs . . . . .	61
4.2. Generalization to new camera-display pairs . . . . .	67
5.1. Generation parameters for A4 and B4 datasets . . . . .	75
5.2. The number of each respective model in the A5 dataset. . . . .	78
5.3. Comparison of classification algorithms . . . . .	86
5.4. Residual network versus siamese network accuracy comparison . . . . .	90



## List of Figures

2.1. QR code on a Times Square Electronic Billboard . . . . .	5
2.2. Image Formation Pipeline . . . . .	6
2.3. Message Recovery Comparison . . . . .	7
2.4. Variance of Light Output among Displays . . . . .	11
2.5. Influence of observation angles . . . . .	12
2.6. Histograms of intensities captured from a uniform display . . . . .	13
2.7. Message Embedding and Retrieval . . . . .	14
2.8. Ratex Patches . . . . .	18
2.9. CDTF Effects on Image Histogram . . . . .	20
3.1. Overview of Differential Metamers . . . . .	29
3.2. MacAdam ellipses for the CIE xy 1931 colorspace . . . . .	30
3.3. Monochromatic images used for differential metamer training . . . . .	36
3.4. Color pairs and separating ellipsoid . . . . .	37
3.5. Set of 14 images used to evaluate BER . . . . .	38
3.6. Learned differential metamers projected into <i>Lab</i> space . . . . .	41
3.7. Graph comparing message recovery across several embedding algorithms	44
4.1. Light Field Messaging (LFM) goal . . . . .	49
4.2. Digital steganography methods such as Baluja [1] are not suitable for photographic steganography . . . . .	50
4.3. CNN architecture . . . . .	53
4.4. Camera-Display 1M examples from 25 camera-display pairs . . . . .	58
4.5. Effect of including $T()$ when training $E()$ and $R()$ . . . . .	62
4.6. Effect of perceptual loss metric on image quality . . . . .	63
4.7. Effect of camera-display viewing angle on message recovery . . . . .	64

4.8. Effects of changing camera exposure . . . . .	65
5.1. Shape from noisy RCS signal overview . . . . .	69
5.2. Four shape families . . . . .	70
5.3. Interclass RCS signal variation . . . . .	70
5.4. Generating monostatic rcs signals from geometric shapes . . . . .	72
5.5. Swerling detectability . . . . .	74
5.6. CNN architecture . . . . .	77
5.7. Siamese network . . . . .	80
5.8. Refiner network . . . . .	82
5.9. Refiner network output . . . . .	83
5.10. Residual network comparison . . . . .	84
5.11. Siamese network confusion matrices . . . . .	85
5.12. Residual network versus siamese networks . . . . .	87
5.13. Three classifier comparison . . . . .	89
5.14. Residual network robustness to distortions . . . . .	92
5.15. Examples of signals pre and post refinement . . . . .	93

# Chapter 1

## Introduction

This thesis describes photographic steganography, the process of imperceptibly encoding messages into photos and video rendered on an electronic display that are decoded using a camera. Several techniques are introduced to increase the accuracy of message recovery, reduce obtrusive perceptual impact, and eliminate synchronization problems between camera and display.

In Chapter 2, we present a novel method for communicating between a moving camera and an electronic display by embedding and recovering hidden, dynamic information within an image. A small intensity pattern is added to alternate frames of a time-varying display. A handheld camera pointed at the display can receive not only the display image, but also an underlying message. Differencing the camera-captured alternate frames leaves the small intensity pattern, but results in errors due to photometric effects that depend on camera pose. Detecting and robustly decoding the message requires careful photometric modeling for message recovery. The key innovation of our approach is an algorithm that performs simultaneous radiometric calibration and message recovery in one convex optimization problem. By modeling the photometry of the system using a camera-display transfer function (CDTF), we derive an *optimal online radiometric calibration (OORC)* for robust computational messaging as demonstrated with nine different commercial cameras and displays. The online radiometric calibration algorithms described in this chapter significantly reduces message recovery errors, especially for low intensity messages and oblique camera angles [2].

In Chapter 3, we exploit human color metamers to send light-modulated messages decipherable by cameras, but camouflaged to human vision. These time-varying messages are concealed in ordinary images and videos. Unlike previous methods which

rely on visually obtrusive intensity modulation, embedding with color reduces visible artifacts. The mismatch in human and camera spectral sensitivity creates a unique opportunity for hidden messaging. Each color pixel in an electronic display image is modified by shifting the base color along a particular color gradient. The challenge is to find the set of color gradients that maximizes camera response and minimizes human response. Our approach does not require a priori measurement of these sensitivity curves. We learn an ellipsoidal partitioning of the 6-dimensional space of base colors and color gradients. This partitioning creates metamer sets defined by the base color of each display pixel and the corresponding color gradient for message encoding. We sample from the learned metamer sets to find optimal color steps for arbitrary base colors. Ordinary displays and cameras are used, so there is no need for high speed cameras or displays. Our primary contribution is a method to map pixels in an arbitrary image to metamer pairs for steganographic camera-display messaging. [3]

The initial work in Chapters 2 and 3 was based on a hand-designed mathematical framework, and suffered from the major problem of needing to know the reference frame. Knowing the reference frame a priori is not practical in real world applications, and sending the reference and embedded frame led to synchronization problems when moving to video rates. In Chapter 4 of this thesis, we have a paradigm shift where we develop a synchronization-free messaging method by learning the camera-display messaging function and discovering the best embedding methodology using deep networks. In Chapter 4, we introduce Light Field Messaging (LFM), a process of embedding, transmitting, and receiving hidden information in video that is displayed on a screen and captured by a handheld camera. The goal of the system is to minimize perceived visual artifacts of the message embedding, while simultaneously maximizing the accuracy of message recovery on the camera side. LFM requires photographic steganography for embedding messages that can be displayed and camera-captured. Unlike digital steganography, the embedding requirements are significantly more challenging due to the combined effect of the screen’s radiometric emittance function, the camera’s sensitivity function, and the camera-display relative geometry. We devise and train a network to jointly learn a deep embedding and recovery algorithm that requires no

multi-frame synchronization. A key novel component is the camera display transfer function (CDTF) to model the camera-display pipeline. To learn this CDTF we introduce a dataset (Camera-Display 1M) of 1,000,000 camera-captured images collected from 25 camera-display pairs. The result of this work is a high-performance real-time LFM system using consumer-grade displays and smartphone cameras [4].

In Chapter 5, we apply modern machine learning techniques to radar signals. Radar systems emit a time-varying signal and measure the response of a radar-reflecting surface. In the case of narrowband, monostatic radar signal domain, all spatial information is projected into a Radar Cross Section (RCS) scalar. We address the challenging problem of determining shape class using monostatic RCS estimates collected as a time series from a rotating object tumbling with unknown motion parameters under detectability limitations and signal noise. Previous shape classification methods have relied on image-like synthetic aperture radar (SAR) or multistatic (multiview) radar configurations with known geometry. Convolutional neural networks (CNNs) have revolutionized learning tasks in the computer vision domain by leveraging images and video rich with high-resolution 2D or 3D spatial information. We show that a feed-forward CNN can be trained to successfully classify object shape using only noisy monostatic RCS signals with unknown motion. We construct datasets containing over 100,000 simulated RCS signals belonging to different shape classes. We introduce deep neural network architectures that produce 2% classification error on testing data. We also introduce a refinement network that transforms simulated signals to appear more realistic and improve training utility. The results are a pioneering step toward the recognition of more complex targets using narrowband, monostatic radar [5]. Chronologically, this work on classifying radar cross section followed our initial steganography methods. The subsequent experiences in deep learning for recognition, led to a rethinking of the the problem of steganography and the development of the Light Field messaging in Chapter 4.

Finally, Chapter 6 concludes with a discussion of solved and unsolved problems in photographic steganography.

## Chapter 2

# Optimal Radiometric Calibration for Camera-Display Messaging

While traditional computer vision concentrates on objects that reflect environment lighting (passive scenes), objects which emit light, such as electronic displays, are increasingly common in modern scenes. Unlike passive scenes, *active scenes* can have intentional information that must be detected and recovered. For example, displays with QR codes [6] can be found in numerous locations such as shop windows and billboards 2.1. However, QR-codes are very simple examples because the bold, static pattern makes detection somewhat trivial. The problem is more challenging when the codes are not visible markers, but are hidden within a displayed image. The displayed image is a light field, and decoding the message is an interesting problem in photometric modeling and computational photography. The paradigm has numerous applications because the electronic display and the camera can act as a communication channel where the display pixels are transmitters and the camera pixels are receivers. Unlike hidden messaging in the digital domain, real-world camera-display messaging is a relatively new area. The problem was introduced with intensity modulation and fixed camera systems [7, 8, 9, 10, 11], and extended to moving cameras [12, 13, 14], high-frequency modulation [15? ], and depth cameras [13]. In this chapter, we develop an optimal method for sending and retrieving hidden time-varying messages using electronic displays and cameras which accounts for the characteristics of light emittance from the display using radiometric calibration. The electronic display has two communication channels: 1) the original display image such as advertising, maps, slides, or artwork; 2) the transmission of hidden time-varying messages.

When light is emitted from a display, the resultant 3D light field has an intensity



Figure 2.1: QR code on a Times Square electronic billboard. The high contrast black-white pattern is relatively easy to detect, track and decode. However, consider the more general task of encoding a message within an unknown and arbitrary image on an electronic display. The detection, tracking and decoding problems become significantly more challenging and interesting. Source: [2].

that depends on the angle of observation as well as the pixel value controlled by the display. The emittance function of the electronic display is analogous to the BRDF (bidirectional reflectance distribution function) of a surface. This function characterizes the light radiating from a display pixel. It has a particular spectral shape that does not match the spectral sensitivity curve of the camera. The effect of the display emittance function, the spectral sensitivity of the camera and the camera viewing angle are all components of our photometric model for image formation as shown in Figure 2.2. Our approach does not require measurement or knowledge of the exact display emittance function. Instead, we estimate the entire system transfer function as a *camera-display transfer function* (CDTF) which determines the captured pixel value as a function of the displayed pixel value. By using online frame-to-frame estimation of the CDTF, no prior calibration is required and the method is independent of the particular choice of display and camera.

Although watermarking literature has many hidden messaging methods, this area

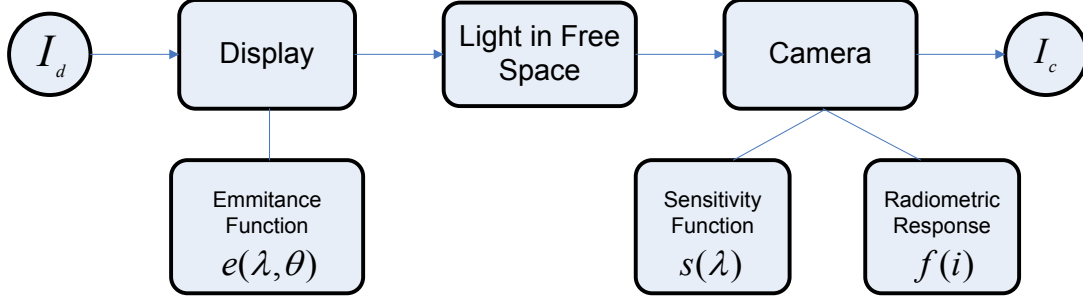


Figure 2.2: Image Formation Pipeline: The image  $I_d$  is displayed by an electronic display with an emittance function  $e$ . The display is observed by a camera with sensitivity  $s$  and radiometric response function  $f$ . Source: [2].

typically ignores the physics of illumination. Display-camera messaging is fundamentally different from watermarking because each pixel of the image is a light source that propagates in free space. Therefore, representations and methods that act only in the digital domain are not sufficient.

The problem of understanding the relationship between the displayed pixel and the captured pixel is closely related to the area of traditional radiometric calibration [16, 17, 18]. In these methods, a brightness transfer function characterizes the relationship between scene radiance and image pixel values. The characterization of this function is done by measuring a range of scene radiances and the corresponding captured image pixels. Our problem in camera-display messaging is similar but has important key differences. The CDTF is more complex than standard radiometric calibration because the system consists of both a display and a camera, each device adding its own nonlinearities. We can exploit the control of pixel intensities on the display and easily capture the full range of input intensities. However, the display emittance function is typically dependent on the display viewing angle. Therefore, the CDTF is dependent on camera pose. In a moving camera system, the CDTF must be estimated per frame; that is, an online CDTF estimation is needed. Furthermore, this function varies spatially over the electronic display surface.

We show that the two-part problem of online radiometric calibration and accurate message retrieval can be structured as an optimization problem. We present an elegant problem formulation where the photometric modeling leads to *physically-motivated*



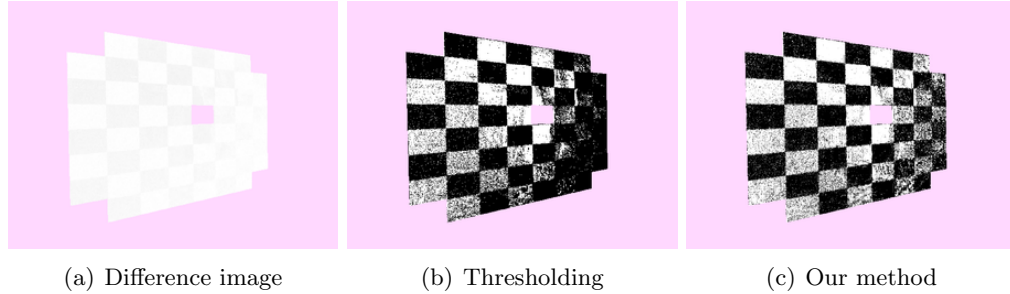


Figure 2.3: Comparison of message recovery with a naive method and the proposed optimal method (a) Difference of two consecutive frames in the captured sequence to reveal the transmitted message. (b) Naive method: Threshold the difference image by a constant (threshold  $T = 5$  for this example). (c) Optimal Method: Bits are classified by a simultaneous radiometric calibration and message recovery. Source: [2].

*kernel functions* that are used with a support vector machine classifier. We show that calibration and message bit classification can be done simultaneously and the resulting optimization algorithm operates in four dimensional space and is convex. The algorithm is a novel method for *online optimal radiometric calibration* (OORC) that enables accurate camera-display messaging. An example message recovery result is shown in Figure 2.3. Our experimental results show that accuracy levels for message recovery can improve from as low as 40-60% to higher than 90% using our approach when compared to either no calibration, or sequential calibration followed message recovery. For evaluation of results, 9 different combinations of displays and cameras are used with 15 different image sequences, for multiple embedded intensity values, and multiple camera-display view angles.

The contributions of the chapter can be summarized as follows: 1) A new optimal online radiometric calibration with simultaneous message recovery, cast as a convex optimization problem; 2) photometric model of the camera display transfer function; 3) the use of ratex (radiometric textured calibration) patches to provide continual calibration information as a practical method for online calibration; 4) the use of distribution-driven intensity mapping as a practical method for visually non-disruptive online calibration.

## 1 Related Work

**Watermarking** In developing a system where cameras and displays can communicate under real world conditions, the initial expectation was that existing watermarking techniques could be used directly. Certainly the work in this field is extensive and has a long history with numerous surveys compiled [19, 20, 21, 22, 23, 24]. Surprisingly, existing methods are not directly applicable to our problem. In the field of watermarking, a fixed image or mark is embedded in an image often with the goal of identifying fraudulent copies of a video, image or document. Existing work emphasizes almost exclusively the digital domain and does not account for the effect of illumination in the image formation process in real world scenes. In the digital domain, neglecting the physics of illumination is quite reasonable; however, for camera-display messaging, illumination plays a central role.

From a computer vision point of view, the imaging process can be divided into two main components: photometry and geometry. The geometric aspects of image formation have been addressed to some extent in the watermarking community, and many techniques have been developed for robustness to geometric changes during the imaging process such as scaling, rotations, translations and general homography transformations [25, 26, 27, 28, 29, 21, 30]. However, the *photometry* of imaging has largely been ignored. The rare mention of photometric effects [31, 32] in the watermarking literature doesn’t define photometry with respect to illumination; instead photometric effects are defined as “lossy compression, denoising, noise addition and lowpass filtering”. In fact, photometric attacks are sometimes defined as jpeg compression [27].

**Radiometric Calibration** Ideally, we consider the pixel-values in a camera image to be a measurement of light incident on the image plane sensor. It is well known that the relationship is typically nonlinear. Radiometric calibration methods have been developed to estimate the camera response function that converts irradiance to pixel values. In measuring a camera response, a series of known brightness values are measured along with the corresponding pixel values. In general, having such ground truth

brightness is quite difficult. The classic method [17] uses multiple exposure values instead. The light intensity on the sensor is a linear function of the time of exposure, so known exposure times enables ground truth light intensity. This exposure-based method is used in several radiometric calibration methods [16, 18, 17, 33, 34]. Our goal for the display-camera system is related to radiometric calibration; the system converts scene radiance to pixels (the camera), but also converts from pixel to scene radiance (the display) so that the whole camera-display system is a function that maps a color value at the display to a color value at the camera.

The camera response in radiometric calibration is either estimated as a full mapping where  $\mathbf{i}_{\text{out}}$  is specified for every  $\mathbf{i}_{\text{in}}$  or as an analytic function  $g(\mathbf{i}_{\text{in}})$ . Several authors [16, 35, 36] use polynomials to model the radiometric response function. Similarly, we have found that fourth order polynomials can be used for modeling the inverse display-camera transfer function. The dependence on color is typically modeled by considering each channel independently [16, 18, 17, 37]. Interestingly, although more complex color models have been developed [38, 39, 40], we have found the independent channel approach suitable for the display-camera representation where the optimality criterion is accurate message recovery.

Existing radiometric calibration methods are developed for cameras, not camera-display systems. Therefore, display emittance function is not part of these prior methods. However, for the camera-display transfer function, this component plays an important role. We do not use the measured display emittance function explicitly, but since the CDTF is view dependent and the camera can move, our approach is to perform radiometric calibration per frame.

**Other Methods for Camera-Display Communication** Camera-display communications have precedent in the computer vision community, but existing methods differ from our proposed approach. For example, researchers on the Bokode project [41] presented a system using an invisible message, however the message is a fixed symbol, not a time-varying message. Invisible QR codes were addressed in [42], but these QR-codes

are fixed. Similarly, traditional watermark approaches typically contained fixed messages. LCD-camera communications is presented in [8] with a time-varying message, but the camera is in a fixed position with respect to the display. Consequently, the electronic display is not detected, tracked or segmented from the background. Furthermore, the transmitted signal is not hidden in this work. Recent work has been done in high speed visible light communications [43], but this work does not utilize existing displays and cameras and requires specialized hardware and LED devices. Time-of-flight cameras have recently been used for phase-based communication [44], but these methods require special hardware. Interest in camera-display messaging is also shared in the mobile communications domain. COBRA, RDCode, and Strata have developed 2D barcode schemes designed to address the challenges of low-resolution and slow shutter speeds typically present in smartphone cameras [45, 46, 47]. Likewise, Lightsync has targeted synchronization challenges with low frequency cameras [48].

## 2 System Properties

In our proposed camera-display communication system, pixel values from the display are inputs, while captured intensities from the camera are output. We denote the mapping from displayed intensities to captured ones as *Camera-Display Transfer Function* (CDTF). In this section, we motivate the need for online radiometric calibration by briefly analyzing factors that influence the CDTF.

**Display Emittance Variation** Displays vary widely in brightness, hue, white balance, contrast and many other parameters that will influence the appearance of light. To affirm this hypothesis, an SLR camera with fixed parameters observes 3 displays and models the CDTF for each one as shown in Figure 2.4. Although each display is tuned to the same parameters, including contrast and RGB values, each display produces a unique CDTF.

**Observation Angles** Electronic displays emit light with an angular dependence. Consider the image of an electronic display captured by a camera from multiple angles

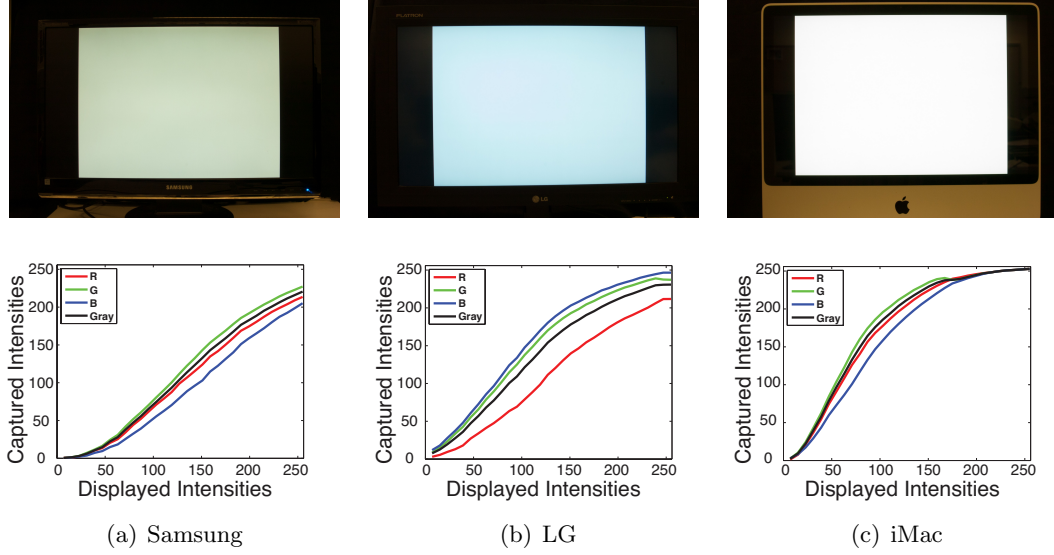


Figure 2.4: **Variance of Light Output among Displays.** An SLR camera captured a range of grayscale  $[0,255]$  intensity values produced by 3 different LCDs. These 3 CDF curves highlight the difference in the light emittance function for different displays. Source: [2].

as shown in Figure 4.7. More oblique observation angles yield lower captured pixel intensities. Additionally, there is a nonlinear relationship between captured light intensity and viewing angle.

### 3 Methods

#### 3.1 Photometry of Display-Camera systems

The captured image  $\mathbf{i}_c$  from the camera viewing the electronic display image  $\mathbf{i}_d$  can be modeled using the image formation pipeline in Figure 2.2. First, consider a particular pixel within the display image  $\mathbf{i}_d$  with red, blue and green components given by  $\boldsymbol{\rho} = (\rho_r, \rho_g, \rho_b)$ . The captured image  $\mathbf{i}_c$  at the camera has three color components  $(i_c^r, i_c^g, i_c^b)$ , however there is no one-to-one correspondence between the color channels of the camera sensitivity function and the electronic display emittance function. When the monitor displays the value  $(\rho_r, \rho_g, \rho_b)$  at a pixel, it is emitting light in a manner governed by its emittance function and modulated by  $\boldsymbol{\rho}$ . The monitor emittance function  $\mathbf{e}$  is typically a function of the viewing angle  $\boldsymbol{\theta} = (\theta_v, \phi_v)$  comprised of a polar and

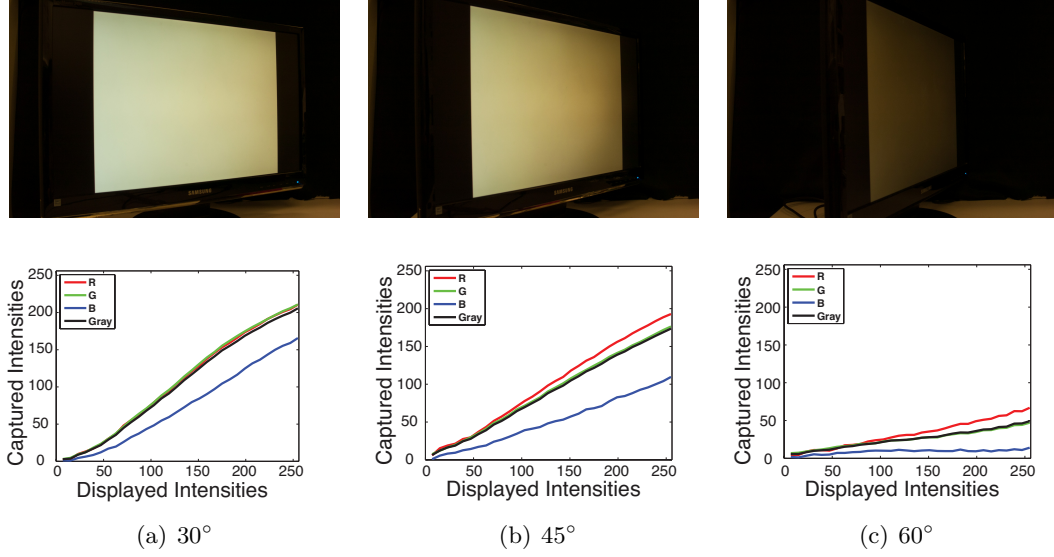


Figure 2.5: **Influence of observation angles.** Using the Nikon-Samsung pair, a range of grayscale  $[0, 255]$  values were displayed and captured from a set of different observation angles. As observation angle became more oblique, the camera-display transfer function changes. Source: [2].

azimuthal component. For example, the emittance function of an LCD monitor has a large decrease in intensity with polar angle (see Figure 2.6).

The emittance function has three components, i.e.  $\mathbf{e} = (e_r, e_g, e_b)$ . Therefore the emitted light  $\mathbf{i}$  as a function of wavelength  $\lambda$  for a given pixel  $(x, y)$  on the electronic display is given by

$$i(x, y, \lambda) = \rho_r e_r(\lambda, \theta) + \rho_g e_g(\lambda, \theta) + \rho_b e_b(\lambda, \theta), \quad (2.1)$$

or

$$i(x, y, \lambda) = \boldsymbol{\rho}^T \mathbf{e}(\lambda, \boldsymbol{\theta}). \quad (2.2)$$

Now consider the intensity of the light received by one pixel element at the camera sensor. Let  $s_r(\lambda)$  denote the camera sensitivity function for the red component, then the red pixel value  $i_c^r$  can be expressed as

$$i_c^r \propto \int_{\lambda} [\boldsymbol{\rho}^T \mathbf{e}(\lambda, \boldsymbol{\theta})] s_r(\lambda) d\lambda. \quad (2.3)$$

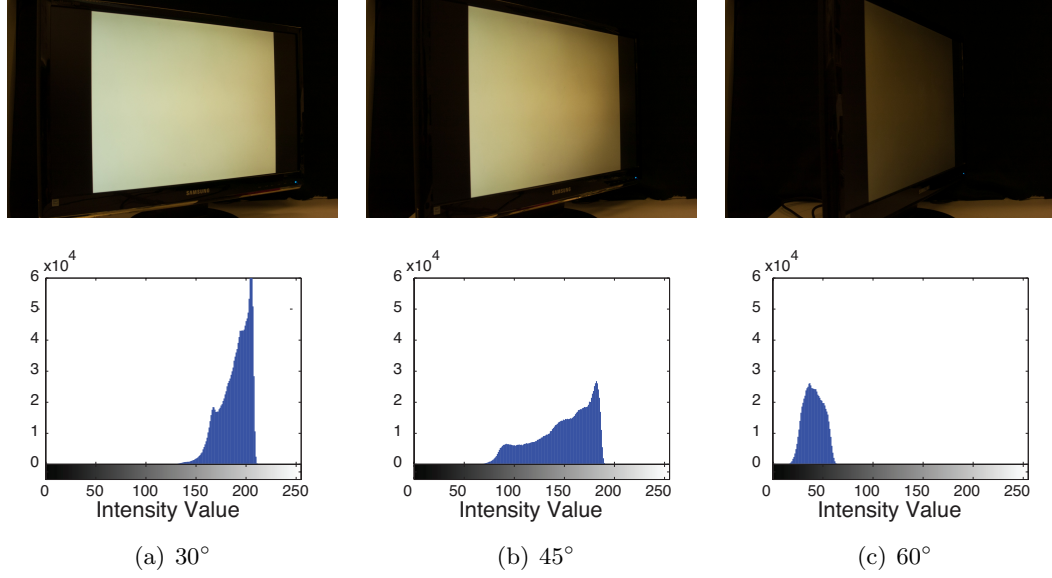


Figure 2.6: **Histograms of intensities captured from a uniform display.** Notice as observation angle changes, so does the distribution of captured intensities illustrating the angular variation of the display emittance function. Source: [2].

Notice that the sensitivity function of the camera has a dependence on wavelength that is likely different than the emittance function of the monitor. That is, the interpretation of “red” in the monitor is different from that of the camera. Notice that a sign of proportionality is used in Equation 2.3 to specify that the pixel value is a linear function of the intensity at the sensor, assuming a linear camera and display. This assumption will be removed in Section 3.3.

Equation 2.3 can be written to consider all color components in the captured image  $\mathbf{i}_c$  as

$$\mathbf{i}_c \propto \int_{\lambda} [\boldsymbol{\rho}^T \mathbf{e}(\lambda, bm\theta)] \mathbf{s}(\lambda) d\lambda. \quad (2.4)$$

where  $\mathbf{s} = (s_r, s_g, s_b)$ .

### 3.2 Message Structure

The pixel value  $\boldsymbol{\rho}$  is controllable by the electronic display driver, and so it provides a mechanism for embedding information. We use two sequential frames in our approach. We modify the monitor intensity by adding the value  $\kappa$  and transmit two consecutive images, one with the added value  $\mathbf{i}_e$  and one image of original intensity  $\mathbf{i}_o$ . To get a

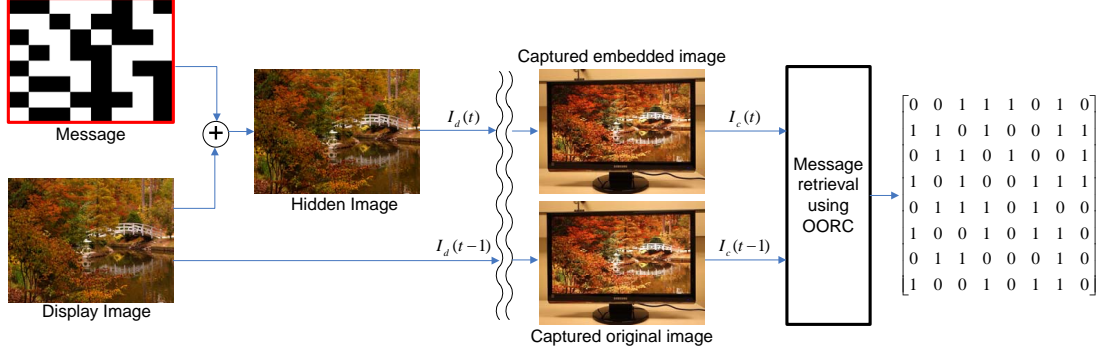


Figure 2.7: Message Embedding and Retrieval. Two sequential frames are sent, an original frame and a frame with an embedded message image. Simple differencing is not sufficient for message retrieval. Our method (OORC) is used to recover messages accurately. Source: [2].

rectangular frontal-view message, a homography warp is applied to the images only after pixel-wise frame subtraction. The recovered message depends on the display emittance function and camera sensitivity function if the embedded message is done by adding  $\kappa$  as follows:

$$\mathbf{i}_e \propto \int_{\lambda} [(\kappa + \boldsymbol{\rho}^T) \mathbf{e}(\lambda, \theta)] \mathbf{s}(\lambda) d\lambda. \quad (2.5)$$

Recovery of the embedded signal leads to a difference equation

$$\mathbf{i}_e - \mathbf{i}_o \propto \int_{\lambda} [(\kappa) \mathbf{e}(\lambda, \theta)] \mathbf{s}(\lambda) d\lambda. \quad (2.6)$$

The dependence on the properties of the display  $\mathbf{e}$  and the spectral sensitivity of the camera  $\mathbf{s}$  remains. We use additive-based messaging, instead of ratio-based methods, because this structure is convenient for convexity of the algorithm as described in Section 3.3.

The main concept for message embedding is illustrated in Figure 2.7. In order to convey many “bits” per image, we divide the image region into a series of block components. Each block can convey a bit “1” or “0”. The blocks corresponding to a “1” contain the added value  $\kappa$  typically set to 3 or 5 gray levels on the  $[0, 255]$  scale, while the zero blocks have no additive component ( $\kappa = 0$ ). The message is recovered by sending the original frame followed by a frame with the embedded message and using



the difference for message recovery. The message can also be added to the coarser scales of a image pyramid decomposition [49], in order to better hide the message within the display image content. The display can be tracked with existing methods [50]. This message structure is decidedly very simple, so the methods presented here can be applied to many message coding schemes.

When accounting for the nonlinearity in the camera and display, we rewrite Equation 2.4 to include the radiometric response function  $f$ ,

$$\mathbf{i}_c = f \left( \int_{\lambda} [\boldsymbol{\rho}^T \mathbf{e}(\lambda, \boldsymbol{\theta})] \mathbf{s}(\lambda) d\lambda \right). \quad (2.7)$$

More concisely,

$$\mathbf{i}_c = f(\mathbf{i}_d), \quad (2.8)$$

and the recovered display intensity is

$$\mathbf{i}_d = f^{-1}(\mathbf{i}_c) = g(\mathbf{i}_c). \quad (2.9)$$

We use polynomials to represent the radiometric inverse function  $g(\mathbf{i})$ . The same inverse function  $g$  is used for all color channels. This simplification of the color problem is justified by the accuracy of the empirical results. As the purpose of the calibration algorithm is to explicitly deal with nonlinear responses, no gamma correction is needed.

### 3.3 Optimal Online Radiometric Calibration

The two goals of message recovery and calibration can be combined to a single problem. While ideal radiometric calibration would provide a captured image that is a linear function of the displayed image, we show that calibrating followed by message recovery only gives a relatively small increase in message accuracy. However, if the two goals are combined into a simultaneous problem we have two benefits: 1) the problem formulation can be done in a convex optimization paradigm with a single global solution and 2) the accuracy increases significantly.

Let  $g(\mathbf{i})$  be the inverse function that is modeled with a fourth order polynomial as

follows

$$g(\mathbf{i}) = a_4 \mathbf{i}^4 + a_3 \mathbf{i}^3 + a_2 \mathbf{i}^2 + a_1 \mathbf{i} + a_0. \quad (2.10)$$

Consider two images frames  $\mathbf{i}_o$ , where  $\mathbf{i}_o$  is the original frame and  $\mathbf{i}_e$  the image frame with the embedded message. Since we are using an additive message embedding, we wish to classify the message bits as either ones or zeros based on the difference image  $\mathbf{i}_o - \mathbf{i}_e$ .

Taking into account the radiometric calibration, we want to classify on the recovered data  $g(\mathbf{i}_o) - g(\mathbf{i}_e)$ . We have found empirically that the inverse function can be modeled by a fourth order polynomial, so that the function to be classified is

$$\begin{aligned} g(\mathbf{i}_o) - g(\mathbf{i}_e) = \\ a_4(\mathbf{i}_o^4 - \mathbf{i}_e^4) + a_3(\mathbf{i}_o^3 - \mathbf{i}_e^3) + a_2(\mathbf{i}_o^2 - \mathbf{i}_e^2) + a_1(\mathbf{i}_o - \mathbf{i}_e). \end{aligned} \quad (2.11)$$

In Equation 2.11, we see that the calibration problem has a physically motivated nonlinear mapping function. That is, we see that the original data  $(\mathbf{i}_o, \mathbf{i}_e)$  can be placed into a higher dimensional space using the nonlinear mapping function  $\Phi$  which maps from a two dimensional space to a four dimensional space as follows

$$\begin{aligned} \Phi(\mathbf{i}_o, \mathbf{i}_e) = \\ \left[ \begin{array}{cccc} (\mathbf{i}_o^4 - \mathbf{i}_e^4) & (\mathbf{i}_o^3 - \mathbf{i}_e^3) & (\mathbf{i}_o^2 - \mathbf{i}_e^2) & (\mathbf{i}_o - \mathbf{i}_e) \end{array} \right]. \end{aligned} \quad (2.12)$$

In this four dimensional space we seek a separating hyperplane between the two classes (one-bits and zero-bits). Our experimental results indicate that these are not separable in lower dimensional space, but the movement to a higher dimensional space enables the separation. Also, the form of that higher dimensional space is physically motivated by the need for radiometric calibration. Therefore our problem becomes a support vector machine classifier where the optimal support vector weights and the calibration parameters are *simultaneously* estimated. That is, we estimate

$$\mathbf{w}^T \mathbf{u} + \mathbf{b}, \quad (2.13)$$

where,  $\mathbf{w} \in \mathbf{R}^4$ ,  $\mathbf{b}$  are the separating hyperplane parameters, and  $\mathbf{u}$  is the input feature vector. Since we want to perform radiometric calibration, the four-dimensional input is given

$$\mathbf{u} = \begin{bmatrix} a_4(\mathbf{i}_o^4 - \mathbf{i}_e^4) & a_3(\mathbf{i}_o^3 - \mathbf{i}_e^3) & a_2(\mathbf{i}_o^2 - \mathbf{i}_e^2) & a_1(\mathbf{i}_o - \mathbf{i}_e) \end{bmatrix}^T. \quad (2.14)$$

Notice that the  $\mathbf{w}^T \mathbf{u} + \mathbf{b}$  is still linear in the coefficients of the inverse radiometric function. These coefficients and the scale factor  $\mathbf{w}$  are estimated simultaneously. We arrive at the important observation that accounting for the CDTF preserves the convexity of the overall optimization problem. The coefficients of the function  $g$  are scaled by  $\mathbf{w}$ , so that calibration and classification can be done *simultaneously*, and convexity of the SVM is preserved. We refer to this method as *optimal online radiometric calibration (OORC)* because it recovers radiometric parameters via convex optimization for each frame.

**Ratex Patches** The standard problem of radiometric calibration is solved by varying exposure so that a range of scene radiance can be measured. For CDTF calibration in a single frame, patches are placed within the display image that have intensity variation over the full range of display brightness values (a linear variation with pixel values from 0 to 255). These radiometric textured calibration patches or *ratex patches* are placed in inconspicuous regions of the display image such as an image corner. The ratex patches are not used as part of the hidden message, but instead provide training data in each frame for the OORC method of CDTF calibration and message recovery. Figure 2.8 shows an example where ratex patches are placed in each of the 4 image corners. Consecutive frames of ratex patches toggle between message bit “1” and message bit “0” to provide training data for both message bits.

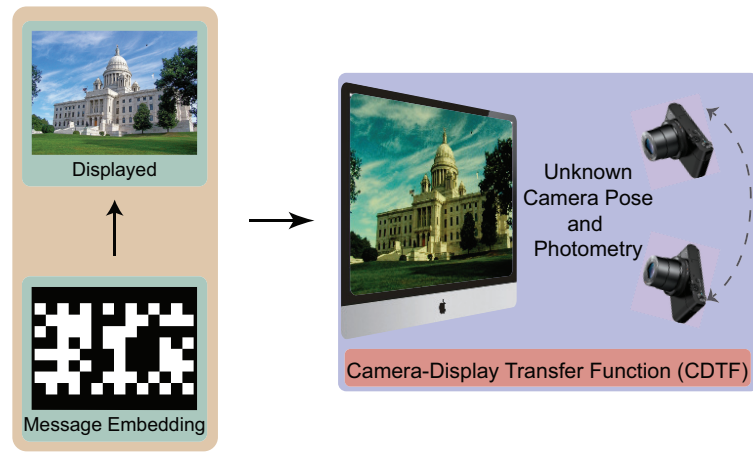


Figure 2.8: Radiometric calibration texture patches (ratex patches). Ratex patches placed in corners and used for radiometric calibration and classification training. Source: [2].

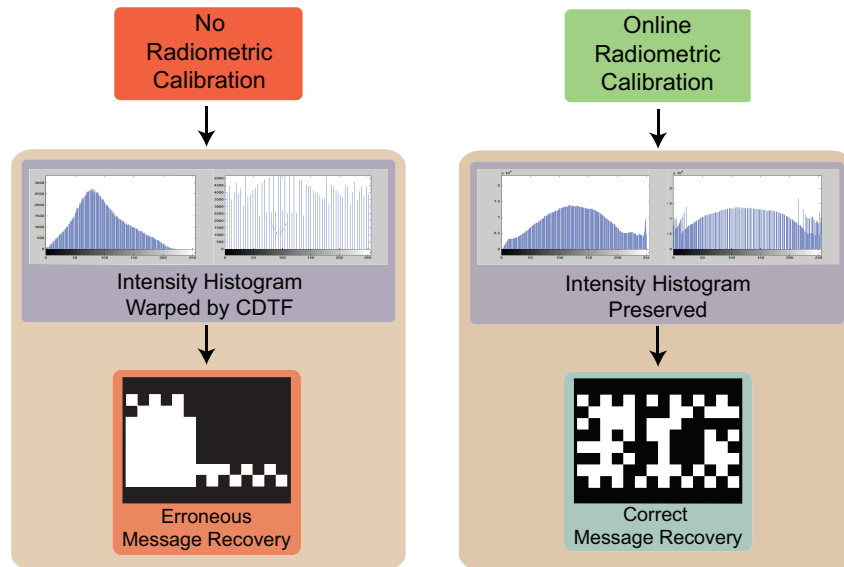
### 3.4 Hidden Ratex

We also introduce a method for radiometric calibration that employs visually non-disruptive *hidden ratex* mapping, since ratex patches can be visually obtrusive and unattractive for applications. Rather than directly measuring the effect that the CDTF has on known intensity values, we measure the effect on the image histogram. Instead of using ratex patches that have a linear variation over the full intensity range, we use display images with intensity values that are well-distributed over the full intensity range. We estimate the CDTF by finding the mapping of the measured histogram to the original histogram. For this approach to work, we need to know the initial intensity histogram of an image before it passes through the CDTF. We perform a simple intensity mapping (equalization) on every image before it is displayed, so the initial intensity histogram is known and uniformly distributed. The camera-captured image is intensity mapped to restore this distribution, after distortion by the CDTF. The inverse CDTF is computed and corrected for in this approach and we refer to this method as *hidden ratex* since no visible patches are used. The hidden ratex method is

illustrated in Figure 2.9.



(a) Hidden messages rely on small intensity variations and are corrupted by this camera-display transfer function (CDTF).



(b) Online Radiometric Calibration mitigates the distorting effects of the CDTF to enable more accurate message recovery.

Figure 2.9: From the display to the camera, the light signal is affected by display photometry, camera pose and camera radiometry. Hidden messages rely on small intensity variations and are corrupted by this camera-display transfer function (CDTF). In each pair of intensity histograms shown above, the left represents an image histogram before passing through the CDTF, and the right represents the histogram after the CDTF. Online Radiometric Calibration mitigates the distorting effects of the CDTF to enable more accurate message recovery. Source: [2].

## 4 Results

For empirical validation, 9 different combination of displays and cameras are used, comprised of 3 displays: 1) LG M3204CCBA 32 inch, 2) Samsung SyncMaster 2494SW, 3) iMac (21.5 inch 2009); and 3 cameras: 1) Canon EOS Rebel XSi, 2) Nikon D70, 3) Sony DSC-RX100. Fifteen 8-bit display images are used. From each display image, we create a display video of 10 frames: 5 frames with the original display images interleaved with 5 images of embedded time-varying messages. An embedded message frame is followed by an original image frame to provide the temporal image pair  $\mathbf{i}_e$  and  $\mathbf{i}_o$ . The display image does not change in the video, only the bits of the message frames. Each message frame has  $8 \times 8 = 64$  blocks used for message bits (with 5 bits used for ratex patches for calibration and classification training data). Considering 5 display images, with 5 message frames and 59 bits per frame results in approximately 1500 message bits. The accuracy for each video is defined as the number of correctly classified bits divided by the total bits embedded and is averaged over all testing videos. The entire test set over all display-camera combinations is approximately 18,000 test bits.

We evaluate 4 methods for embedded message recovery. Method 1 (*Naive Threshold*) has no radiometric calibration, only the difference  $\mathbf{i}_e - \mathbf{i}_o$  is used to recover the message bit via thresholding. Method 2 (*Two-step*) is radiometric calibration using ratex patches followed by thresholding the interframe difference  $\mathbf{i}_e - \mathbf{i}_o$  for message recovery. Method 3 (*OORC*) is the optimal calibration where both radiometric calibration and message recovery are done simultaneously. Method 4 *Hidden Ratex* is calibration using hidden ratex intensity mapping followed by simple differencing for message recovery. The methods we introduced here (Methods 2-4) demonstrate significant improvement over naive thresholding. For methods 2 and 3, training data from pixels in the ratex patches are used to train an SVM classifier. For method 4, no visible patches are needed. For each of the 9 display-camera combinations, the accuracy of the 4 message recovery methods was tested with 2 sets of experimental variables: 1)  $0^\circ$  frontal camera-display view; 2)  $45^\circ$  oblique camera-display view; and: 1) embedded message intensity difference

of 5; 2) embedded message intensity difference of 3. The results of these tests are can be found in Tables 2.1, 2.2, 2.3, and 2.4. Notice that naive thresholding has low message recover rates (as low as 47.5% for oblique views). Message recovery rates were highest for the OORC method with recovery rates of 98-99% for most camera display combinations even for oblique views. The hidden ratex method also maintained near 90% recognition rates for oblique views and had the advantage of having no visible calibration patches.

Accuracy (%)	Naive Thresh-old	Two-step	OORC	Hidden Ratex
Canon-iMac	72.94	75.67	99.17	89.63
Canon-LG	58.94	84.94	98.44	95.74
Canon-Samsung	48.44	64.89	99.39	89.91
Nikon-iMac	60.17	75.50	95.17	90.00
Nikon-LG	49.72	73.39	99.33	94.81
Nikon-Samsung	47.22	72.89	95.00	89.54
Sony-iMac	64.44	76.00	99.06	71.11
Sony-LG	56.11	75.61	98.56	90.93
Sony-Samsung	47.50	79.11	98.89	87.80
Average	56.17	75.33	98.11	88.83

Table 2.1: This table shows our main result. Accuracy of embedded message recovery and labeling with additive intensity  $\kappa = +3$  on  $[0,255]$  and captured with  $45^\circ$  oblique view. Low  $\kappa$  values are desirable (because they are less noticeable) but lead to larger errors, especially at oblique views. Our calibration methods can greatly increase accuracy (from 47-50% to over 90% ) in some cases.

## 5 Discussion

This chapter identifies many of the challenges associated with imperceptible camera-display messaging. We jointly model the display emittance function, camera sensitivity function, and radiometric effects of light in free space as the camera-display transfer function (CDTF). We show that naive thresholding, while intuitively simple, is a poor



Accuracy (%)	Naive Threshold	Two-step	OORC	Hidden Ratex
Canon-iMac	85.56	83.06	96.44	91.57
Canon-LG	86.39	90.94	98.67	94.07
Canon-Samsung	87.94	87.78	98.94	91.30
Nikon-iMac	84.06	84.00	96.50	90.27
Nikon-LG	74.67	81.44	99.94	90.09
Nikon-Samsung	77.33	86.06	98.00	91.57
Sony-iMac	89.33	84.22	99.44	70.00
Sony-LG	87.61	95.39	99.72	80.74
Sony-Samsung	80.00	83.78	96.26	84.54
Average	83.56	86.30	98.22	87.13

Table 2.2: Accuracy of embedded message recovery and labeling with additive intensity  $\kappa = +3$  on  $[0,255]$  and captured at  $0^\circ$  frontal view.

choice because the variation of display intensity with camera pose is ignored. These methods lead to lower message recovery rates, especially for oblique views ( $45^\circ$ ) and small intensity messages. We introduce two methods for online radiometric calibration for camera-display messaging. The first method, Optimal Online Radiometric Calibration (OORC), yields the best message recovery accuracy, but requires visually obtrusive ratex patches to be placed in the corners of the image. The second method, Hidden Ratex, uses histogram equalization to outperform naive thresholding without visually obtrusive ratex patches, but does not outperform OORC in terms of message recovery accuracy. We demonstrate experimental results for nine different camera-display combinations at frontal and  $45^\circ$  oblique viewing directions.

The results indicate a marked improvement in message recovery over naive thresholding for camera-display messaging with our methods. Prior methods of digital watermarking do not take into account the photometric effects of the camera-display transfer

Accuracy (%)	Naive Threshold	Two-step	OORC	Hidden Ratex
Canon-iMac	97.06	94.50	99.83	95.37
Canon-LG	87.89	99.00	99.39	99.44
Canon-Samsung	71.67	88.11	100.00	95.37
Nikon-iMac	91.89	93.67	96.00	96.11
Nikon-LG	81.56	95.11	99.94	98.88
Nikon-Samsung	58.78	92.22	99.39	97.41
Sony-iMac	92.28	92.00	99.72	80.37
Sony-LG	77.06	96.22	100.00	91.13
Sony-Samsung	63.28	94.17	99.89	81.67
Average	80.16	93.89	99.35	93.71

Table 2.3: Accuracy of embedded message recovery and labeling with additive intensity  $\kappa = +5$  on  $[0,255]$  and captured with  $45^\circ$  oblique perspective.

function and the resulting dependence on camera pose. Therefore these prior methods are likewise prone to error. Our experimental results show that hidden, dynamic messages can be embedded in a display image and recovered robustly.

Accuracy (%)	Naive Threshold	Two-step	OORC	Hidden Ratex
Canon-iMac	95.28	96.61	99.00	95.74
Canon-LG	97.11	99.72	97.17	97.59
Canon-Samsung	97.39	97.33	98.94	94.35
Nikon-iMac	98.39	99.17	99.22	96.11
Nikon-LG	99.83	100.00	99.83	97.31
Nikon-Samsung	96.33	97.44	98.56	95.74
Sony-iMac	97.72	97.00	99.94	81.67
Sony-LG	99.39	100.00	100.00	90.74
Sony-Samsung	92.50	92.33	98.06	90.28
Average	97.10	97.73	98.97	93.28

Table 2.4: Accuracy of embedded message recovery and labeling with additive intensity  $\kappa = +5$  on  $[0,255]$  and captured at  $0^\circ$  frontal view. The problem is relatively straightforward for this case with frontal views and high  $\kappa$  value (5). The benefits of radiometric calibration are much more apparent in Tables 2.1, 2.2,2.3, where errors are larger when the  $\kappa$  value is decreased, and for oblique views.

## Chapter 3

# Reading Between the Pixels: Photographic Steganography for Camera-Display Communication

### 1 Introduction

Electronic displays, such as LCD monitors, are typically used only for human visual observation. Research in the relatively new field of camera-display communication has introduced a dual channel: a machine-readable communications channel operating in parallel with the human-observable display. Time-varying messages can be embedded in the on-screen images, but this task has significant challenges. The modulated signal is an illumination field propagating in free-space, so prior methods of watermarking for digital images are not directly applicable. The illumination field emitted by the display and captured by the camera depends on the parameters of the radiometric transfer function and sensitivity curves of both the display and camera. This camera-display transfer function makes message recovery challenging, but it also presents an opportunity for message embedding that is tuned to typical transfer functions.

A common method for camera-display messaging relies on intensity modulation either for directly embedding bits or for embedding transformation coefficients [2, 15]. Human vision is generally very sensitive to intensity step edges, even when the step size is small. For simple messaging, the display image can be modified by adding a message image where “1” bit values are encoded in a block by a small intensity step and “0” bit values are encoded by zero intensity step. The message frame is added in alternative temporal frames so that sequential frame subtraction can be used to decode the message. This method assumes that the display image is constant over time intervals. Accurate message recovery is challenging because small intensity steps are needed to hide the message, but large intensity steps are needed for a low-noise

signal that can be accurately decoded by the camera. To avoid cross-talk between the machine-readable and human-readable channels, other methods rely on infrared, high speed equipment, or low-throughput encoding schemes.

Another approach to making the message imperceptible is to use high speed light modulation so that the flicker fusion effect of human vision can temporally blur the intensity variation [51]. In Nguyen et al., information is sent from display to camera using high speed intensity modulation and adaptive codes that hide spacial modulation in highly-textured areas of a carrier image. These methods provide a correct throughput of 22 kbps while remaining hidden to the user. High speed displays and cameras are commercially available, but the higher cost is prohibitive for ubiquity in electronic signage and mobile display applications. Since this method is based on intensity modulation, it could easily be modified for color modulation, further reducing the amount of noticeable flicker and further improving bit recovery rate.

Our approach uses *color modulation* that exploits the differences in human color sensitivity versus camera color sensitivity. This allows us to accurately send and receive camouflaged messages without specialized hardware. In a displayed image  $\mathbf{i}$ , let the pixel coordinate be denoted by  $\mathbf{w} \in \mathbf{R}^3$ . Each image pixel  $\mathbf{i}(\mathbf{w})$  has 3 color components,  $\mathbf{i}(\mathbf{w}) \in \mathbf{R}^3$ . A color message image  $\mathbf{m}$  is added to  $\mathbf{i}$  such that our steganographically embedded image  $\mathbf{e} = \mathbf{i} + \mathbf{m}$ , and a pixel of the embedded message is given by  $\mathbf{e}(\mathbf{w}) = \mathbf{i}(\mathbf{w}) + \mathbf{m}(\mathbf{w})$ . For “1” bits, the message  $\mathbf{m}$  is a color shift added to  $\mathbf{i}$ . The goal is to find the best color shift  $\boldsymbol{\delta} \in \mathbf{R}^3$ . Let  $\hat{\boldsymbol{\delta}}$  denoted the unit direction in color space, and  $\|\boldsymbol{\delta}\|_2$  is the magnitude of the step-size. We seek a step  $\boldsymbol{\delta}$  to create a *differential metamer*  $\mathbf{g} = (\mathbf{i}(\mathbf{w}), \mathbf{i}(\mathbf{w}) + \boldsymbol{\delta})$  such that  $\mathbf{i}(\mathbf{w}) + \boldsymbol{\delta}$  is perceived to be the same color as  $\mathbf{i}(\mathbf{w})$  by a human observer but is camera-captured as a distinguishable color, where  $\mathbf{g} \in \mathbf{R}^6$ .

Large sets of differential metamers can be generated given a small training set. Our approach uses a 6-dimensional quadratic binary classifier, solved in a convex optimization problem. Using training data with positive and negative examples, the algorithm determines a set of separating ellipsoids in 6-dimensional space, an example of this is shown in Figure 3.4. The interior of these ellipsoids contain 6-dimensional points  $\mathbf{g}$  where the first three components corresponds to a particular base color and the last

three components provide the corresponding  $\hat{\delta}$  used for messaging. The interior of these 6-dimensional ellipsoids define approximate metamer sets that sufficiently provide message hiding and recovery.

## Differential Metamers

Traditionally, *metamers* are colors that have different spectral power distributions, but appear identical to observers when integrated over the 3 cones sensitivities in the human eye (see Figure 3.2). We introduce the term *differential metamers* to define pairs of color values programmed for sequential display that result in minimal visible change for the human observer but are distinguishable colors when captured by a camera. This process is illustrated in Figure 3.1. Many differential metamers exist even among 8-bit color values, but finding the color values that yield both low human sensitivity and high camera sensitivity is difficult because  $256^6$  (over  $2 \times 10^{14}$ ) is the number of potential metamer pairs to be tested for both camera-display sensitivity and human-display sensitivity. Specific camera sensitivity curves combined with human vision parameters are not be enough to model the differential metamer space. Display parameters indicating the spectrum of light emission for each programmed color vector and the dependence on radiometric observation parameters are also be needed to determine an analytical model. Given the variations involved, we choose a data driven approach instead. We show that this approach is straightforward and effective. We generate samples in 6D space indicating base colors and color gradients for messaging. By observation of the resulting messaging visibility (human and camera), these sample points are labeled as “good” or “bad” for messaging. By sampling 2480 points, we train a set of ellipsoidal binary classifiers that predict successful differential metamers where the base color values  $\mathbf{i}$  fill the displayable color space. We perform the metamer set estimation in both RGB and CIE *Lab* color spaces.

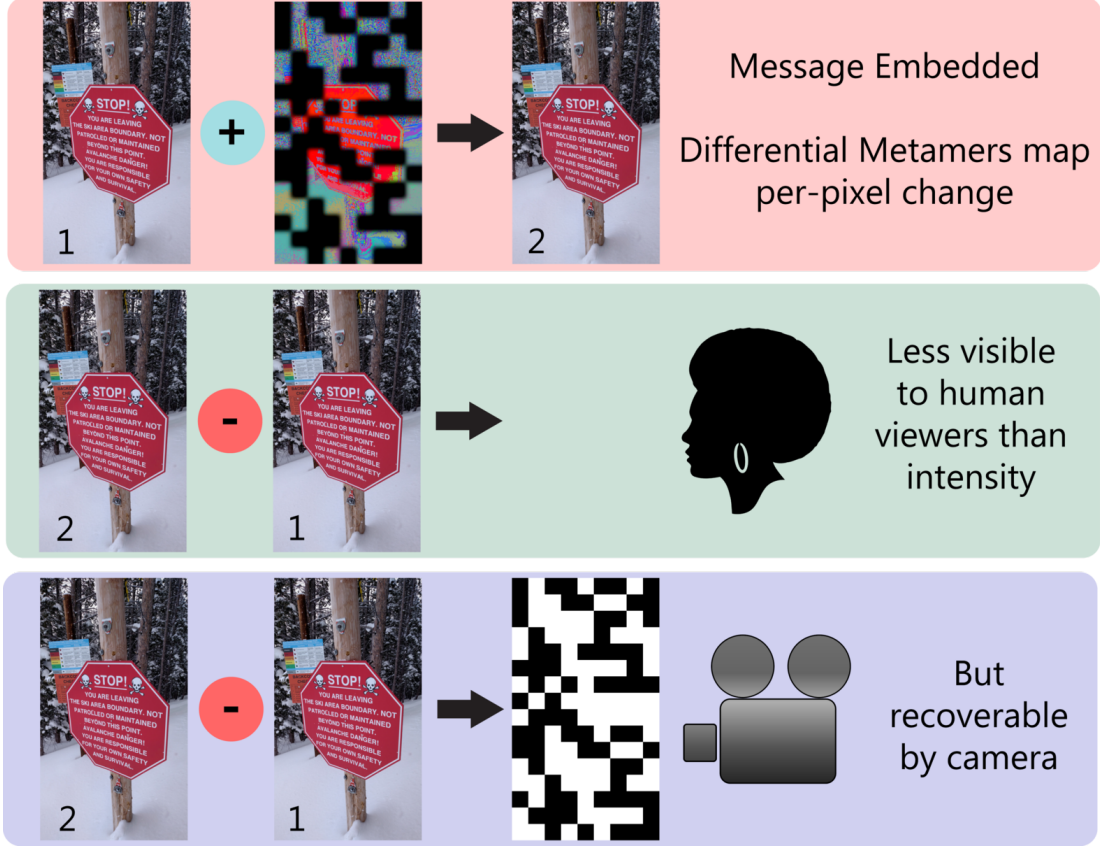


Figure 3.1: Differential Metamers are color pairs that are optimized to be identical under human vision, but distinguishable by a camera. By modulating small per-pixel changes within an image sequence, differential metamers can be used to embed hidden messages. The embedded message is blended to reduce the spatial visibility without disturbing camera recovery.

## 2 Background and Related Work

**Metamers and Separating Ellipsoids** Our approach to finding separating ellipsoids in color space is motivated by two main factors. First, the problem of fitting a separating ellipsoid to labeled data is a convex optimization problem [54] and therefore is not affected by local minima. Second, human vision research has showed the utility of ellipsoidal surface fitting for representing color difference thresholds. As early as the 1940's, human vision studies identified and quantified ellipsoidal representations for the problem of understanding human sensitivity to small color differences [55, 52] as illustrated in Figure 3.2. This ellipsoidal representation has been confirmed in numerous studies in early vision literature [56, 57, 58]. Parametric surfaces were used to

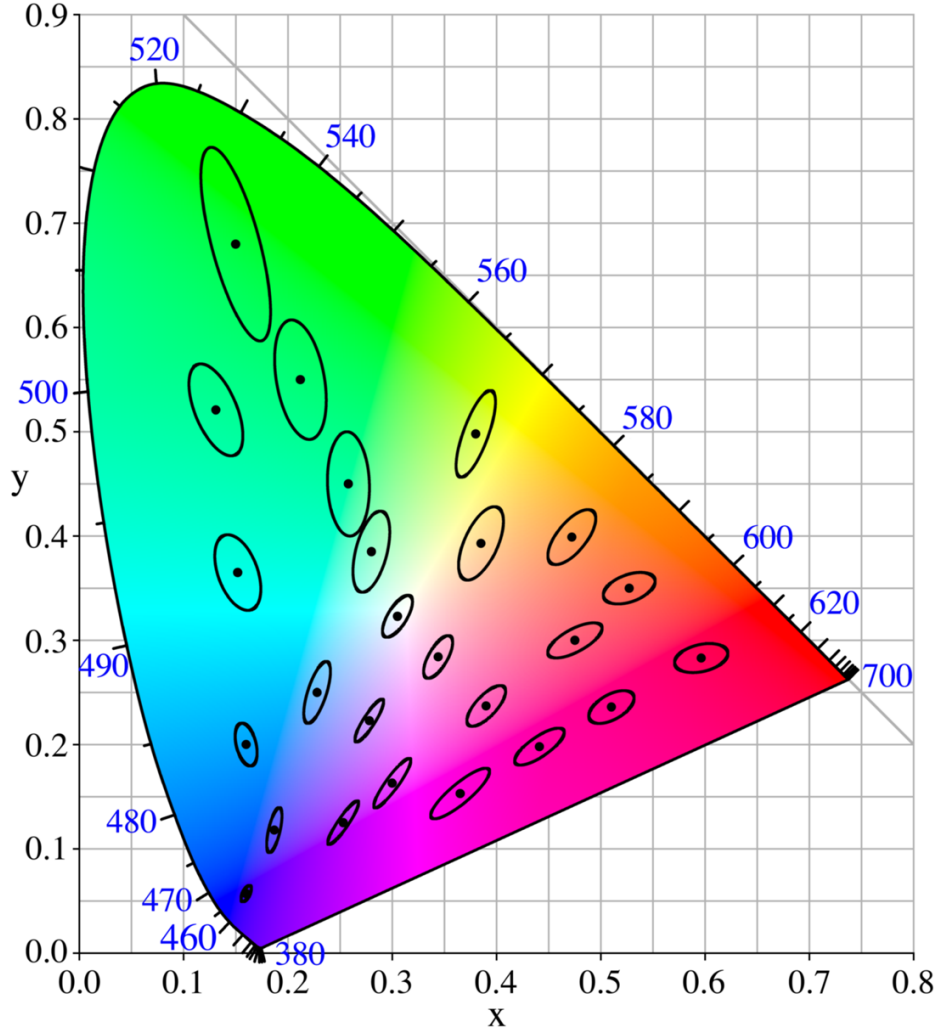


Figure 3.2: MacAdam ellipses for the CIE xy 1931 colorspace [52, 53]. The area within these scaled-up ellipses represent metamers, or colors which cannot be distinguished.

find discriminating contours. The fitting typically used detection thresholds [59, 60] in order to get just-noticeable-difference (JND) contours [61]. Our framework greatly simplifies this process because no threshold values are measured. Instead, a set of learned separating ellipsoids finds a discriminatory boundary between color pairs that are differential metamers and those that are not. Metamer sets [62] are well described by ellipsoids [52]. By extension, we have adopted discriminating ellipsoids to characterize the space of differential metamers. In prior work that used color to embed information [63] color gradients are used to watermark spatially varying microstructures into images. The objective in this work is to embed watermarks that were difficult to see



from a distance, but visible up close. This is different from our goal of finding pairs of colors where no distinction can be made when viewed sequentially by humans, but the difference can be robustly detected by a camera.

**Camera-Display Communication** Electronic displays such as televisions, computer monitors, and projectors are traditionally used to display images, videos, and text - all human readable scenes. These devices can also display camera-readable images such as QR-codes [2, 47, 45, 44, 64, 10, 65, 8, 15, 14, 11, 66, 7]. Within the past 5 years, extensive work has been done to expand the capabilities of camera display messaging by increasing throughput.

PixNet introduced Orthogonal Frequency-Domain Multiplexing (OFDM) transmission algorithms to address the unique characteristics of the camera-display link, including perspective distortion, blur, and sensitivity to ambient light [8]. While PixNet offer impressive data throughput, it can only display machine-readable code and supports no hybrid approach. Strata introduced distance-scalable coding schemes [47], preferable in a mobile application, but also cannot display both human-readable and camera readable images at the same time. Both of the aforementioned techniques encode bit values with intensity. COBRA introduced a 2D color code [45], but also could only display machine readable code.

Both Visual MIMO [2, 64, 10, 11, 66, 7] and HiLight [65] use intensity modulation in human-readable images to embed a second machine-readable channel. However, it is well known that human vision is extremely sensitive to temporal and spatial changes in intensity. It has been shown that intensity changes, even with small magnitude are likely to cause flicker and discomfort to a human observer. The amount of human visual obtrusion had not been measured for either method.

Kaleido [67] and VRCodes [68] uses metamers to embed data in alternating pixel values. These values, however, are not “true” metamers in the sense that two static colors have different physical properties such as wavelength, but appear identical to human viewers. Instead, Kaleido and VRCodes leverage flicker fusion to create temporally blended colors hidden from human observers with high speed changes. This

approach is constrained by the need for specialized high-speed displays and cameras. VRCodes also leverages the rolling shutter camera typically found on mobile phones to sample at frequencies above 60Hz. Unfortunately, this limits VRCode throughput to only 1 bit per frame.

Kaleido [67] attempts to solve a different problem: embedding noise with flicker fusion metamers to disrupt piracy via camera recording of videos, while preserving the human-visible channel. While similar in intuition to the work presented in this chapter, the goals are fundamentally different. We embed camera-sensitive information in this invisible channel, while Kaleido only embeds camera-sensitive noise. And as stated before, Kaleido requires specialized high-speed displays, while our method requires no specialized hardware.

LED arrays have used modulated light to communicate [11, 69, 14]. Recently, LED-based communication techniques have used color-shift keying for communication [70]. Methods exist to make this color-shift keying imperceptible to human observers [71], but these applications do not require the imperceptible reproduction of high resolution images.

In this work, we take a data driven approach to generating differential metamers that have a small human sensitivity gradient, but large camera sensitivity gradient. We show that differential metamers are effective for steganographically embedding messages into high-quality images on electronic displays.

### 3 Photographic Steganography System Design

**Embedding Steganographic Messages** The message structure we employ is a 2D barcode grid, 16 blocks wide and 9 blocks tall, containing 144 bits in total. The barcode spans the entire display area. To reduce the visible artifacts from sharp spatial gradients, the block pattern is blended. The dimensions of the 2D barcode were chosen empirically. With smaller blocks, more bits can be transmitted in a single image. But as spatial redundancy is reduced, bit recovery errors will increase. Messages larger than 144 bits can be constructed by stringing together sequential 144-bit messages. For each block, a color shift keys a “1” bit. No change to the base color keys a “0” bit.

We represent a differential metamer as the 6-dimensional vector  $\mathbf{g}$  separated into two components  $\mathbf{g} = [\mathbf{g}_b \ \mathbf{g}_m]^T$  where  $\mathbf{g}_b$  is the base color in *Lab* space with  $\mathbf{g}_b \in \mathbf{R}^3$  and  $\mathbf{g}_m$  is the optimal color shift  $\boldsymbol{\delta} \in \mathbf{R}^3$  in the same color space.

The core problem is finding the optimal  $\boldsymbol{\delta}$  for an arbitrary pixel base color. We denote  $\mathbf{G}$  as a set of differential metamers. For each pixel coordinate  $\mathbf{w}$ , we compute the minimum distance between  $\mathbf{i}(\mathbf{w})$  and  $\mathbf{g}_b$  for every member of  $\mathbf{G}$ . We refer to the  $\mathbf{g}$  with the nearest  $\mathbf{g}_b$  as  $\mathbf{g}^*$ , and  $\mathbf{g}_m^*$  provides the corresponding color shift for  $\mathbf{i}(\mathbf{w})$ . So if  $\mathbf{i}(\mathbf{w})$  belongs to a block keyed with a “1” bit, then  $\mathbf{e}(\mathbf{w}) = \mathbf{i}(\mathbf{w}) + \boldsymbol{\delta}$ .

When the images  $\mathbf{i}$  and  $\mathbf{e}$  are rendered, they are transformed by the display’s spectral emittance function  $D()$  which is unknown. When the images are displayed in a video sequence, odd frames display the original image  $D(\mathbf{i})$ , and even frames display the steganographically embedded image  $D(\mathbf{e})$ .

**Recovering Steganographic Messages** The two image frames are sequentially imaged by the camera. The displayed images are affected by light travel in free space and are transformed by the camera’s spectral sensitivity function. Denote these two unknown transformation functions  $F()$  and  $C()$  respectively. The camera-captured images  $C(F(D(\mathbf{i})))$  and  $C(F(D(\mathbf{e})))$  are subtracted from each other. For each bit-block, an average difference greater than some threshold corresponds to a “1”, and below that threshold corresponds to a “0”. The threshold is calculated by reserving 4 of the 144 bits for calibration. The recovered message was then compared to the known message to calculate BER (bit error rate). BER is the percentage of misclassified bits in each 144 bit message.

$$\text{BER} = \frac{\text{count( incorrectly classified bits )}}{\text{count( all bits )}},$$

## 4 Learning New Differential Metamers

As stated in Section 1, differential metamers exist even among 8-bit color values. But testing  $256^6$  colors is expensive and undesirable. Our approach for generating an expanded gamut of differential metamers relies on a training set of base colors  $\mathbf{i}(\mathbf{w})$  and color shift gradients  $\boldsymbol{\delta}$ . Positive examples in this training set meet the criteria for

embedding: no visible flicker and accurate camera recovery. Negative examples do not meet the criteria for embedding: color pairs that are either visible when viewed sequentially or not recoverable by the camera.

The data resides in 6-dimensional space  $\mathbf{R}^6$ . We choose the number of separating ellipsoids  $k$  empirically and cluster the positive examples into  $k$  clusters in  $\mathbf{R}^6$ . For each cluster, we use convex optimization to find the optimal ellipsoid that separates positive from negative data. Sampling within the union of all separating ellipsoids reveals a dense set of new differential metamers.

For each cluster  $k_i$ , the optimal separating ellipsoid is found. Each ellipsoid separates the positive training examples in cluster  $k_i$  from all negative training examples.

## Collecting and Labeling Training Data

The set of 124 base colors are generated by uniformly sampling CIE *Lab* space. For each base color, 20 baricentrically sampled unit vectors are generated. In total, we now have 2480 training examples.

The algorithm for finding differential metamers has three main components:

1. Cluster positive training examples into  $k$  clusters.
2. For each cluster, find the optimal ellipsoid that separates positive and negative data.
3. Sample within the union of all ellipsoids to find new differential metamers.

### 4.1 Clustering Training Data

A single ellipsoid does not reasonably represent the set of all differential metamers, because color shift is dependent on base color. Therefore we define a separating ellipsoid for each cluster of training data. The positive training points are clustered into  $k$  clusters. The number of clusters is defined as  $k = 50$ , which was chosen after empirical evaluation and performing kernel density estimation.

We compare the recovery error for several embedding algorithms across several step-sizes. The magnitude of the color step size is defined as the L-2 Norm:

$$\|\delta\|_2 = \sqrt{\delta_L^2 + \delta_a^2 + \delta_b^2} \quad (3.1)$$

Recovery error is defined as bit error rate (BER). A diverse set of 14 different images were used to test BER as shown in Figure 4.4. The camera used is a Basler acA2040-90uc-CVM4000, and the display used is an Acer S240HL IPS (in-plane switching) LCD monitor.

A video sequence is generated. Odd frames consist of only a monochromatic image of the base color. Even frames comprise the base color plus a 2D barcode grid corresponding to a message. For these tests, the same checkerboard message is used every time, since it maximizes spatial variation and is likely to be noticed by humans. Examples of this video sequence are shown in Figure 3.3. A camera views the 2480 image sequences only once and attempts to recover the embedded messages. The camera is fixed 0.5 meters from the display with a viewing angle normal to the image plane.

For each of the 2480 training examples, human participants were shown video sequences each containing a single color and with an embedded checkerboard pattern alternating at 8Hz for 10 seconds. 8Hz was chosen because humans are particularly sensitive to intensity changes at this frequency [72], and because it represents a reasonable target for smartphone video capture rates. The participants were asked to indicate if they could see the checkerboard pattern or not. Three participants were used for human vision evaluation. They were students between ages 19 and 24. One participant wore glasses, and none had any color-blindness. The variance in their flicker labeling was negligible.

Single-color, monochromatic images are used to isolate the exact behavior of each color pair, and negating the cloaking effects of image content (e.g. texture) and preventing participants from confusing the effects of other, nearby pixels. Relative contrast may have an effect on visibility in real images, but this can be overcome by embedding differential metamers only in a select subset of pixels, or by first clustering nearby pixels

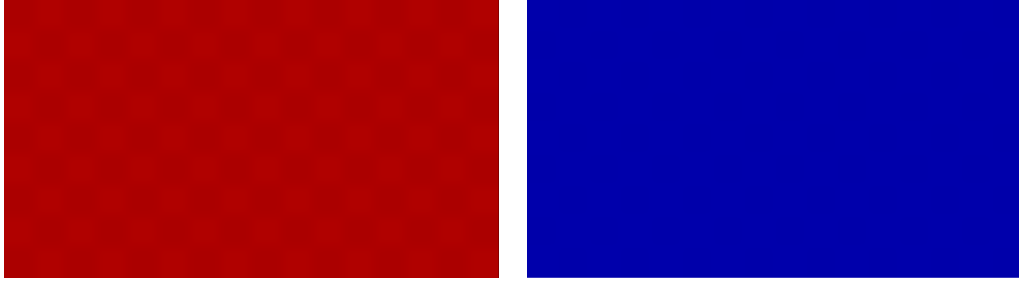


Figure 3.3: Monochromatic images with embedded barcode messages are used for differential metamer training. Images like these are shown to human observers to test whether they can see an embedded checkerboard. These same images are evaluated with a camera-display pair to test if the message is robustly recoverable. The checkerboard is visible in the leftmost image and would be labeled a negative example, while the checkerboard is not visible in the rightmost image and would be labeled a positive example.

by differential metamer gradients and not embedding on the cluster borders. While an evaluation of spatial obtrusiveness caused by relative contrast is interesting, it is outside the scope of this chapter and left for future work.

Positive training examples are defined as ones whose color embedding were completely invisible to humans, but recoverable by camera with BER (bit error rate) = 0%. All other examples were labeled negative training data. After labeling, 922 positive and 1558 negative examples were used for training. Examples of positive and negative pairs are shown in Figure 3.3

### Learning $k$ Optimal Separating Ellipsoids

We have two sets of points in  $\mathbf{R}^6$ ,  $\{\mathbf{x}_1, \dots, \mathbf{x}_N\}$  and  $\{\mathbf{y}_1, \dots, \mathbf{y}_M\}$ . The points  $\mathbf{x}_i$  represent the base colors and modulation steps that satisfy the requirements for embedding: BER = 0%, and no visible flicker. While the points  $\mathbf{y}_i$  do not satisfy both of these conditions. We wish to find a function  $f : \mathbf{R}^n \rightarrow \mathbf{R}$  that is positive on the first set, and negative on the second, *i.e.*,

$$f(\mathbf{x}_i) > 0, \quad i = 1, \dots, N, \quad f(\mathbf{y}_i) < 0, \quad i = 1, \dots, M. \quad (3.2)$$

When these inequalities hold, we say that  $f$  separates the two sets of points.

Cluster of Training Data: Base Colors and Deltas in Lab color space

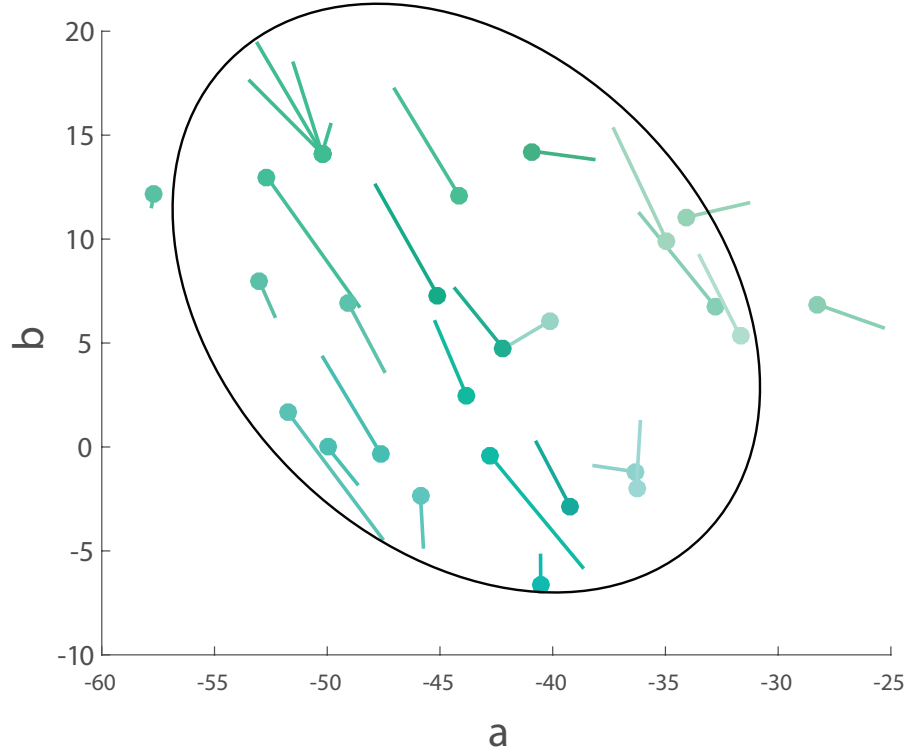


Figure 3.4: A separating ellipsoid and single cluster of positively labeled training data. Visualizing  $k$  6-dimensional ellipsoids is difficult, so the data has been projected down to 2D  $a$   $b$  space (from  $Lab$  color space). We show base colors and color shifts at the same time. The solid circle represents the base colors  $\mathbf{i}$ , and its respective line segments represents the color shift gradients  $\delta$ . The color of each circle and line segment is the actual base color. Notice how there is a general axis of color shift direction for the data in this cluster. Since these are positive training examples, this indicates that human viewers are relatively insensitive to these color shifts. This also indicates that our camera is sensitive to these color shifts.

**Quadratic Discrimination** Since our data points cannot be separated by a  $N$ -dimensional hyperplane, we seek classification via nonlinear discrimination. As long as the parameters that define  $f$  are linear (or affine), the above inequality can still be solved with convex optimization.

In this case, we choose  $f$  to be quadratic and in homogeneous form:

$$f(\mathbf{z}) = \mathbf{z}^T \mathbf{P} \mathbf{z} + \mathbf{q}^T \mathbf{z} + r, \quad (3.3)$$

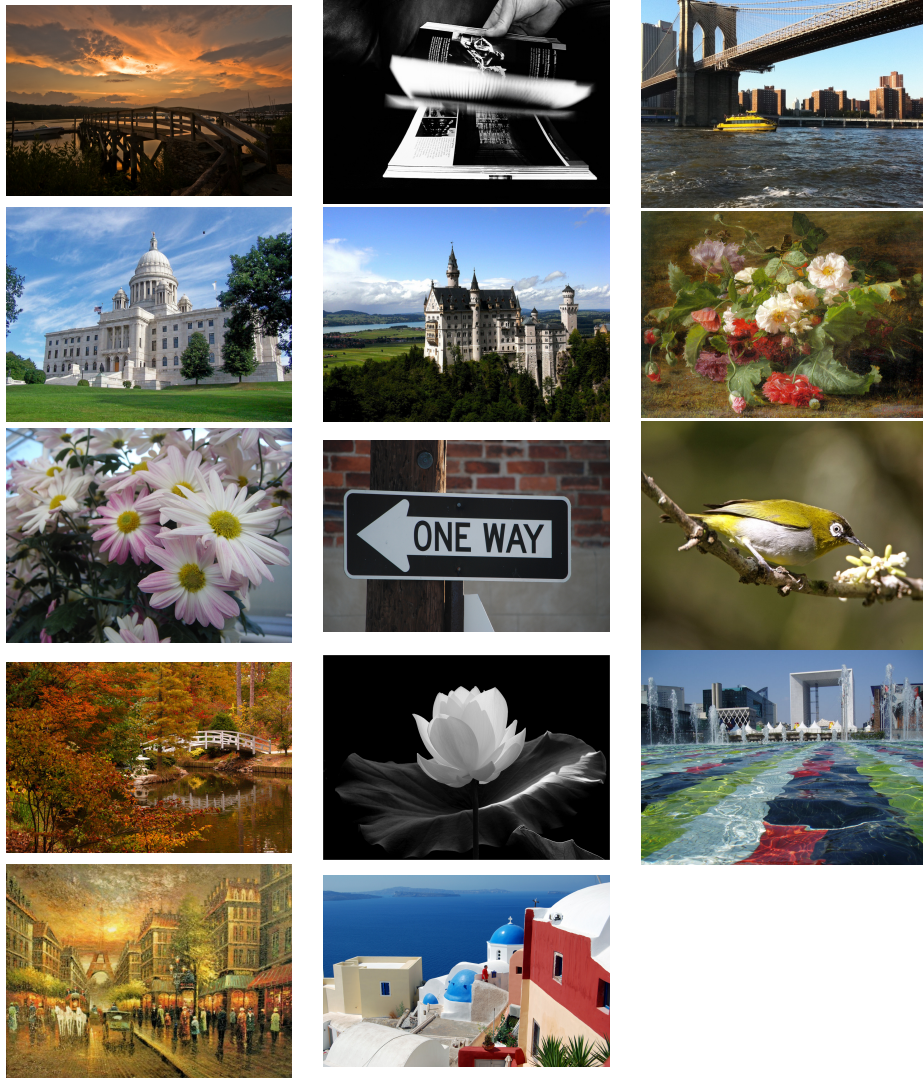


Figure 3.5: Set of 14 images used to evaluate BER across several embedding algorithms and message step-sizes.

where  $\mathbf{P} \in \mathbf{S}^n$  ( $P$  is a symmetric  $n \times n$  matrix),  $\mathbf{q} \in \mathbf{R}^n$ , and  $r \in \mathbf{R}$ , with dimensionality  $n = 6$ . Those parameters  $\mathbf{P}$ ,  $\mathbf{q}$ ,  $r$  are bound by the following constraints:

$$\begin{aligned} \mathbf{x}_i^T \mathbf{P} \mathbf{x}_i + \mathbf{q}^T \mathbf{x}_i + r &> 0, & i = 1, \dots, N \\ \mathbf{y}_i^T \mathbf{P} \mathbf{y}_i + \mathbf{q}^T \mathbf{y}_i + r &< 0, & i = 1, \dots, M \end{aligned} \tag{3.4}$$

Next, we replace 0 with  $\epsilon$ , creating a separating band that is  $2\epsilon$  wide:



$$\begin{aligned}
\mathbf{x}_i^T \mathbf{P} \mathbf{x}_i + \mathbf{q}^T \mathbf{x}_i + r &\geq \epsilon, & i = 1, \dots, N \\
\mathbf{y}_i^T \mathbf{P} \mathbf{y}_i + \mathbf{q}^T \mathbf{y}_i + r &\leq -\epsilon, & i = 1, \dots, M
\end{aligned} \tag{3.5}$$

Dividing out by  $\epsilon$  and subsuming the scalar  $\frac{1}{\epsilon}$  into  $\mathbf{P}, \mathbf{q}, r$ , you arrive at Eq. 3.6. Following [54], we solve for the parameters  $\mathbf{P}, \mathbf{q}, r$  by solving the non-strict feasibility problem:

$$\begin{aligned}
\mathbf{x}_i^T \mathbf{P} \mathbf{x}_i + \mathbf{q}^T \mathbf{x}_i + r &\geq 1, & i = 1, \dots, N \\
\mathbf{y}_i^T \mathbf{P} \mathbf{y}_i + \mathbf{q}^T \mathbf{y}_i + r &\leq -1, & i = 1, \dots, M
\end{aligned} \tag{3.6}$$

The resulting separating surface  $\{\mathbf{z} \mid \mathbf{z}^T \mathbf{P} \mathbf{z} + \mathbf{q}^T \mathbf{z} + r = 0\}$  is quadratic.

**Separating Ellipsoids** We can change the shape of our quadratic separating surface by imposing additional constraints on the parameters  $\mathbf{P}, \mathbf{q}$ , and  $r$ . We form an ellipsoid that contains all points  $\mathbf{x}_1, \dots, \mathbf{x}_N$  but none of the points  $\mathbf{y}_1, \dots, \mathbf{y}_M$  by requiring that  $\mathbf{P} \prec 0$ , that is  $\mathbf{P}$  is negative definite. We can use homogeneity in  $\mathbf{P}, \mathbf{q}, r$  to express the constraint  $\mathbf{P} \prec 0$  as  $\mathbf{P} \preceq -\mathbf{I}$ . We can then cast our quadratic discrimination problem as the following semi-definite programming (SDP) feasibility problem:

$$\begin{aligned}
&\text{find} && \mathbf{P}, \mathbf{q}, r \\
&\text{subject to} && \mathbf{x}_i^T \mathbf{P} \mathbf{x}_i + \mathbf{q}^T \mathbf{x}_i + r \geq 1, & i = 1, \dots, N \\
&&& \mathbf{y}_i^T \mathbf{P} \mathbf{y}_i + \mathbf{q}^T \mathbf{y}_i + r \leq -1, & i = 1, \dots, M \\
&&& \mathbf{P} \preceq -\mathbf{I}
\end{aligned} \tag{3.7}$$

While technically correct, this optimization problem will fail if any of the training points fall outside their classification boundaries. Following the development in [54] for support vector classifiers, we relax our constraints by introducing non-negative variables  $u_1, \dots, u_N$  and  $v_1, \dots, v_M$ . With the relaxation variables  $u_i$  and  $v_i$  introduced, our

inequalities become:

$$\begin{aligned} \mathbf{x}_i^T \mathbf{P} \mathbf{x}_i + \mathbf{q}^T \mathbf{x}_i + r &\geq 1 - u_i, & i = 1, \dots, N \\ \mathbf{y}_i^T \mathbf{P} \mathbf{y}_i + \mathbf{q}^T \mathbf{y}_i + r &\leq v_i - 1, & i = 1, \dots, M \end{aligned} \quad (3.8)$$

The relaxation variables  $u_i$  and  $v_i$  represent the distances of each point outside it's proper boundary. In the original problem,  $u = v = 0$ . We can think of  $u_i$  as a measure of how much each constraint  $\mathbf{x}_i^T \mathbf{P} \mathbf{x}_i + \mathbf{q}^T \mathbf{x}_i + r \geq 1$  is being violated and that's what we want to minimize. A good heuristic is minimizing the sum of variables  $u_i$  and  $v_i$ . The separating ellipsoid defined by  $\mathbf{P}$ ,  $\mathbf{q}$ ,  $r$  is found with the following optimization problem:

$$\begin{aligned} \text{minimize} \quad & \mathbf{1}^T u + \mathbf{1}^T v \\ \text{subject to} \quad & \mathbf{x}_i^T \mathbf{P} \mathbf{x}_i + \mathbf{q}^T \mathbf{x}_i + r \geq 1 - u_i, & i = 1, \dots, N \\ & \mathbf{y}_i^T \mathbf{P} \mathbf{y}_i + \mathbf{q}^T \mathbf{y}_i + r \leq v_i - 1, & i = 1, \dots, M \\ & \mathbf{P} \preceq -\mathbf{I} \\ & \mathbf{u} \succeq 0, \quad \mathbf{v} \succeq 0 \end{aligned} \quad (3.9)$$

To solve this problem we used CVX, a package for specifying and solving convex programs [73, 74]. After each ellipsoid is solved, we test that the ellipsoid is populated before accepting it.

**Sampling Within Union of Ellipsoids** Once  $k$  optimal separating ellipsoids are trained, the points inside the ellipsoids reflect desirable values for message embedding. So to expand our gamut of differential metamers, we densely sample inside the ellipsoid region for new points.  $\mathbf{G}'$  is the expanded set of newly generated differential metamers  $\mathbf{g}'$ . Newly sampled base colors are shown in Figure 3.6. Samples of differential metamer pairs within an ellipsoid are illustrated in Figure 3.4.

**New Lab Differential Metamers generated within 50 seperating ellipsoids**

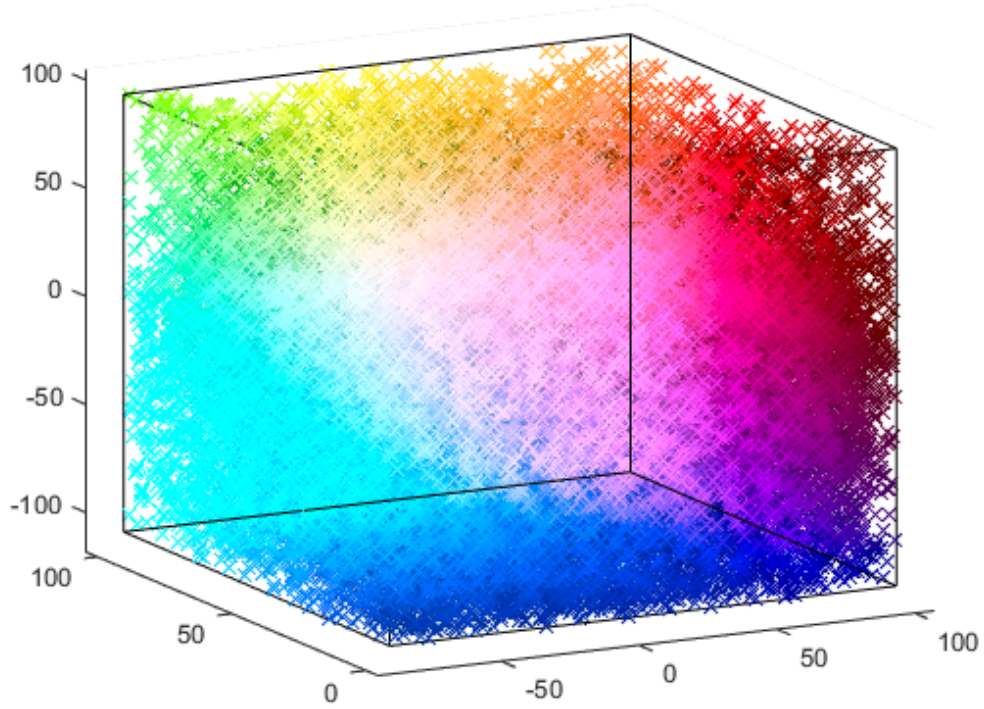


Figure 3.6: Six-dimensional differential metamers are projected down to *Lab* space. These differential metamers are generated by sampling within the separating ellipsoids. Here, the entire *Lab* space is collectively covered by several ellipsoids.

## 5 Experiments

We wish to evaluate the expanded set of differential metamers learned using the techniques described in Section 4. For each of our embedding algorithms, a known message was embedded into a pair of 2 images. A camera then sequentially captured the original image, then the image with the embedded message pattern. Again, the camera was a Basler acA2040-90uc-CVM4000, and the display was an Acer S240HL IPS LCD monitor. The camera was stationed approximately 0.5 meters from the electronic display. The camera had a fixed shutter speed, ISO sensitivity, aperture, and white balance. Each algorithm was evaluated based on the accuracy of recovering each bit of the message. A wide range of message step-sizes were tested. Message step-size refers to the  $\|\delta\|_2$ , or  $\delta$  *magnitude* in 8-bit pixel values. A diverse set of 14 host images was used, shown in Figure 4.4.

For the intensity-based approach, a uniform grayscale  $\delta$  is applied to every pixel representing a “1” bit. The random approach applies a  $\delta$  in a random direction to each pixel. The RGB differential metamers approach assigns a specialized  $\delta$  value to each pixel in the base image. The differential metamer ellipsoids are trained in RGB space. Similarly, the *Lab* differential metamers approach assigns  $\delta$  values from ellipsoids trained in *Lab* space.

## Evaluation of Clustering Methods

Clustering Algorithm	Low Exposure Mean Error	Low Exposure STD	High Exposure Mean Error	High Exposure STD	Runtime ( <i>sec</i> )
<i>k</i> -Means	30.41%	10.65%	24.16%	11.08%	0.0435
<i>k</i> -Medoids	28.97%	10.89%	22.97%	9.92%	0.5813
Gaussian Mixture Models	27.33%	10.83%	22.72%	11.05%	0.0978
Hierarchical clustering	29.37%	11.42%	22.97%	11.64%	0.1299
Spectral clustering	34.52%	11.58%	24.70%	9.91%	0.1387

Table 3.1: Camera recovery error for various clustering methods (*lower is better*). Gaussian Mixture Models (GMMs) produce results with the lowest average errors under both exposure conditions with a small margin of success. In this case, GMMs provide an adequate balance of runtime cost and performance.

A series of clustering algorithms were evaluated: kmeans, kmediods, Gaussian Mixture Models, Hierarchical clustering, and Spectral clustering. Ellipsoids were trained and learned using each of these clustering methods. The ellipsoids yielded differential metamers used for steganographic embedding and recovery. This evaluation is performed twice for each clustering algorithm under two different illumination conditions. Once where the camera has fixed high-exposure settings, and once again with fixed low-exposure settings.

The respective mean errors were 27.285%, 25.97%, 25.025%, 26.17%, and 29.61%. The respective run times were 0.0435s, 0.5813s, 0.0978s, 0.1299s, and 0.1387s. Gaussian Mixture Models (GMMs) yielded the lowest BER on average. Although the margin of superiority is practically nothing, Gaussian mixture models are chosen as the best balance of error and run-time. This study shows that the choice of clustering algorithm has practically no effect on the BER.

Embedding Algorithms				
$\ \delta\ _2$	Intensity	Random	<i>RGB</i> Differential Metamers	<i>Lab</i> Differential Metamers
1	50.69%	50.99%	50.45%	49.85%
2	47.92%	48.81%	42.06%	42.06%
3	43.85%	46.97%	36.11%	37.25%
4	37.00%	44.59%	29.02%	27.83%
5	34.52%	42.41%	22.42%	21.73%
6	23.41%	41.22%	19.84%	17.61%
7	18.70%	38.10%	15.53%	15.08%
8	13.49%	35.57%	13.84%	12.80%
9	09.97%	34.72%	12.50%	12.00%
10	09.13%	32.89%	11.01%	10.91%

Table 3.2: BER for various embedding schemes (*lower is better*). The red-shaded cells indicate  $\delta$  magnitudes where an blended message pattern is easily visible. The green-shaded cells indicate optimal values where the blended message pattern is camouflaged from human vision, but in a good position to be camera-recovered. Differential metamers generated with trained ellipsoids in CIE *Lab* are especially effective because both the BER is reduced and the threshold for acceptable step-size is increased. Notice that for a mid-range step-size of 5 or 6, the *Lab* differential metamers significantly outperform intensity modulation.

Regardless of method used or illumination condition, the standard deviation hovered around 10% for all methods. This suggests that the recovery error results are largely dependent on the base image used. This result has been verified empirically as well; certain images produce better embedding results. The run time calculations took place on an Intel 6700K processor with 32 GB of memory running Matlab 2015b.

## 6 Results

Table 4.1 shows the average message recovery for each embedding algorithm across a variety of  $\|\delta\|_2$  values (step sizes). The red-shaded cells represent values for which the  $\|\delta\|_2$  is so large, the message pattern can be obviously detected by humans. Figure 3.7 illustrates these results graphically.

For small  $\|\delta\|_2$ , the RGB and *Lab* differential metamer approaches greatly outperform the alternatives. Small step sizes are typically preferable because they are more difficult for humans to see. With the differential metamer approach, larger step size can

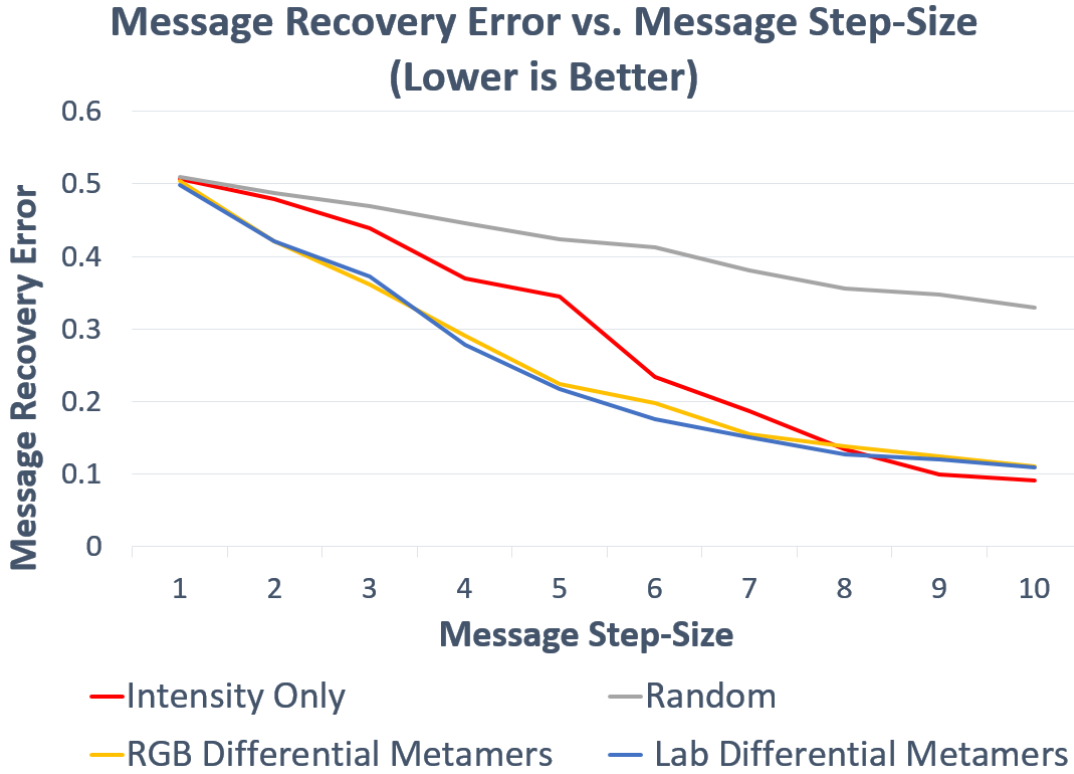


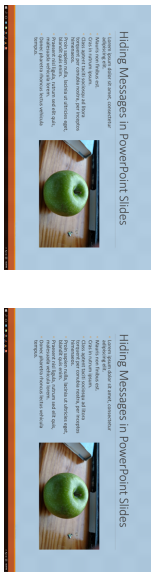

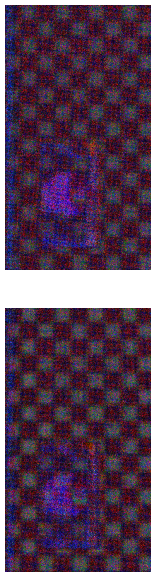
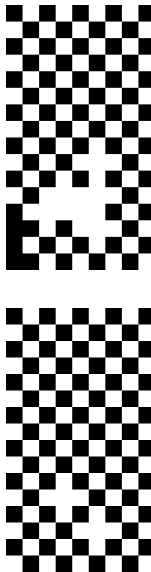
Figure 3.7: This graph compares message recovery across several embedding algorithms. Regardless of embedding algorithm, as message step size ( $\|\delta\|_2$ ) increases, message recovery error decreases. However, large step size also means a more visually obtrusive embedding. For an embedded message to be invisible, smaller step size are greatly preferred. For small to mid-range  $\|\delta\|_2$ , color embedding using differential metamers is significantly better.

be used, facilitating more accurate camera recovery. The differential metamers trained in *Lab* space are most effective at reducing human detection with most robust message recovery. Table 3.3 illustrates these results.

Although the mean error is high compared to perfect recovery, it can be functionally reduced using error-correcting codes. The proposed color messaging framework is applicable to more sophisticated photo-steganographic messaging systems. For the purposes of this chapter, only the reduction in error due to color messaging is evaluated.

### Transferring Learned Ellipsoids to New Hardware

The results presented thus far showcase the effectiveness of photographic steganography using differential metamers trained on a single camera-display pair. But we want to

Low Texture Image:			
Intensity vs CIE Lab Differential Metamers			
Intensity	Differential Metamers		
Image with Embedded Message			
			
Per-pixel difference			
			
Camera-recovered difference			
			
Recovered Message			
			
BER (lower is better)		1.39%	
5.56%			

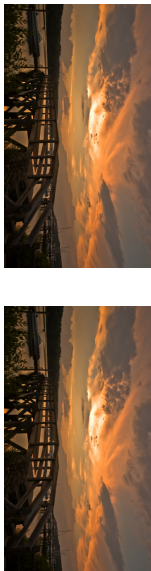

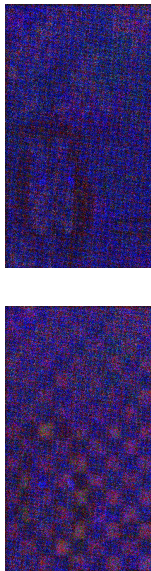

Highly Textured Image:			
Intensity vs CIE Lab Differential Metamers			
Intensity	Differential Metamers		
Image with Embedded Message			
			
Per-pixel difference			
			
Camera-recovered difference			
			
Recovered Message			
			
BER (lower is better)		19.44%	
38.19%			

Table 3.3: Message embedding with intensity vs differential metamers example. The image in the first row contains a steganographic message pattern. Below that, the per-pixel difference shows the ground truth of exactly the changes that were made to the original image. The camera-recovered difference shows the difference measured after the image has been displayed electronically, and captured by a camera. Notice that the differences between ground truth and camera-captured are large. Embedding messages with *Lab* differential metamers is effective for many types of images, including slide or sign type images, as is shown in (a). The example in (b) showcases a more challenging natural image case, where intensity embedding fails in dark and highly textured areas of the image. *Lab* differential metamers are significantly more effective for robust message embedding and recovery. In both (a) and (b),  $\|\delta\|_2 = 5$  for all algorithms.

know how well our learned ellipsoids will transfer to a new camera-display pair. If new differential metamers must be learned for every camera-display combination, the applicability of our algorithm is limited. Table 3.4 features experimental results when the camera-display pair used for photographic steganography is totally different from the camera-display pair used for training. Although the illumination conditions and imaging pipeline remain unchanged, the most significant aspects of the system have been changed. When using different hardware, the BER increases by only 3.48%. Using the same hardware, transferred differential metamers significantly outperform intensity-based embedding. The differential metamers learned under certain hardware conditions can be transferred for a small accuracy cost. Messages with error-correcting codes tolerant of smaller signal-to-noise ratios (SNR) should be incorporated when transferring learned differential metamers to new hardware.

## 7 Discussion

In this chapter, we present a color modulation method used to steganographically embed messages into ordinary images and videos. We develop a data-driven approach to learn a pixel mapping function that produces an optimal differential metamer pair for any pixel value. These differential metamers are pairs of color values that minimize human visual response, but maximize camera response. The key innovation is a novel color-selection framework that leverages the mismatch between human spectral and camera sensitivity curves. We refer to this task of camouflaged camera-display messaging as photographic steganography.

We demonstrate the effectiveness of our differential metamer generation algorithm with message embedding. The goal is to maximize throughput, minimize recovery error, and camouflage the visible artifacts to humans. Although the BER results shown in Table 4.1 are relatively large, message recovery can be significantly improved using radiometric calibration methods, as discussed in [2].

The desirability of our approach stems from the creation of a communication side-channel without using specialized hardware. Embedded information could be used to grant access that is *conditioned on close physical presence* for security or convenience.



Unlike NFC (near-field communications) which is commonly used for precise location verification but has problems with network saturation for nodes in close proximity, beacons using photographic steganography would ensure that users are facing a particular direction. For example, users would not be able to access a networked projector unless they used photographic steganography to recover a dynamic access code embedded in the projectors displayed images to prove that they are in the appropriate location. Scenarios include those where users perform scavenger-hunt games in museums or use outdoor electronic billboards for tickets/coupons/schedules. It is also easy to envision a scenario where users install a smartphone application and have access to extra content on live-broadcast videos.

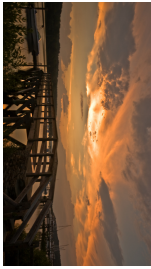
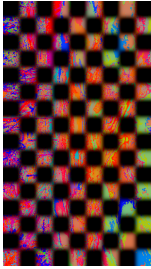
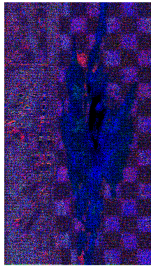
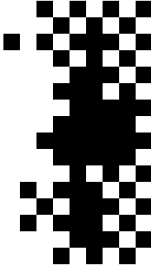
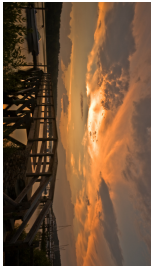
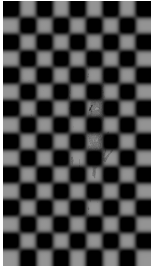
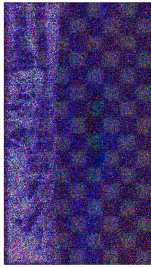

Transferring Learned Ellipsoids to a New Camera-Display Pair				
Transferred Differential Metamers				
Image with Embedded Message	Per-pixel difference	Camera-recovered difference	Recovered Message	
				
Transferred Differential Metamers BER = 22.92% (lower is better)				
Intensity Embedding				
Image with Embedded Message	Per-pixel difference	Camera-recovered difference	Recovered Message	
				
Intensity BER = 36.11% (lower is better)				

Table 3.4: Photographic Steganography using differential metamers learned with a different camera-display pair. An Acer Predator MNT XB271HUC IPS display and Basler acA1300-30uc camera were used in experiment. However, the ellipsoids yielding differential metamers were trained using the aforementioned Basler acA2040-90uc-CVM4000 camera and Acer S240HL display. With  $\|\delta\|_2 = 5$ , the recovered message has a BER of 22.92%, only 3.48% worse than the hardware used for training as shown in Table 3.3. This example demonstrates that the ellipsoids learned can be robustly transferred between different hardware and still significantly outperform intensity-based embedding.

## Chapter 4

### Light Field Messaging

#### 1 Introduction



Figure 4.1: Goal of LFM (Light Field Messaging): embed a message within an image or video, display the image/video on-screen, photograph it with a handheld camera, and recover the hidden message. LFM significantly outperforms other synchronization-free steganography techniques for camera-display messaging in message bit recovery error (BER). Source: [4]. Our code and dataset are available here [75].

In Light Field Messaging (LFM), cameras receive hidden messages from electronic displays concealed within ordinary images and videos. There are many applications for visually concealed information including interactive visual media, augmented reality, road signage for self-driving cars, hidden tags for robotics, privacy-preserving communication, and tagged digital artwork. When the hidden message is recovered from on-screen images, the task has significant challenges and is fundamentally different from the traditional task of steganography. The conversion of a digital image into a light field depends on the characteristics of the electronic display such as the spectral emittance function and spatial emitter pattern. Similarly, the transformation of light field to image depends on the camera pose, sensitivity curves, spatial sampling, and radiometric response. Our unique approach is to learn the entire pathway as a single camera-display transfer function (CDTF) modeled by a supervised deep network. This CDTF component is then used in a larger network that maximizes the accuracy of the camera-recovered message, while minimizing the perceived artifacts in the observed

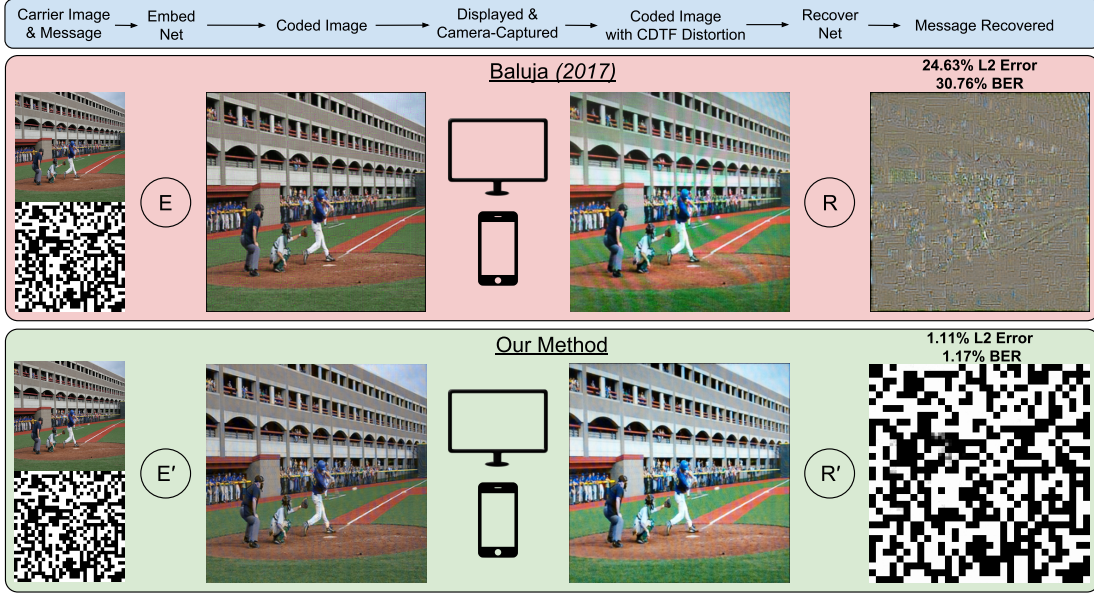


Figure 4.2: Digital steganography methods such as Baluja [1] are not suitable for photographic steganography. The distorting effect of the light field transfer, as characterized by the camera-display transfer function (CDTF), destroys the information steganographically encoded in carrier image pixels. We compare the digital steganography methods introduced by Baluja (top) with our proposed photographic steganography method (bottom). Unlike previous methods, the proposed method includes a model of the CDTF within the training pipeline so that a learned steganographic function for embedding and recovery is robust to CDTF distortion. Source: [4].

display image.

Electronic displays emit light in free space and capturing this *light field* has long been a topic of interest in the computer vision and computer graphics community [76]. Since the camera must capture the signal from the light field, instead of a direct digital path, we use the term *light field messaging*. When the display uses a hidden message we use the term *photographic steganography* to indicate the both the hidden nature and the recovery method using camera-based photography.

Steganography in prior years referred almost exclusively to the digital domain where images are processed and transferred as digital signals [77]. The classic methods for *digital steganography* range from simple alteration of least significant intensity bits to more sophisticated fixed-filter transform domain techniques [19]. Recent work has moved the prior fixed filter approaches to incorporate modern deep learning [1]; but these methods are designed for digital steganography and fail completely for the task of light field

messaging as illustrated in Figure 4.2.

In this chapter, we propose a single-shot end-to-end *photographic steganography* algorithm for light field messaging. Our method is comprised of: a CDTF network to model the camera and display without radiometric calibration; an embedding network to optimally embed the message within an image; and a message recovery network to retrieve the message on the camera side. A major advantage of our approach is single-frame operation so that no temporal synchronization between camera and display is needed, greatly increasing the practical utility of the method. Synchronization is a major issue, and the results of Chapters 2 and 3 were done with pre-synchronized image pairs, not video frames. We assume that properties of the camera hardware, display hardware, and radiometry are not known beforehand. Instead, we develop a training dataset *Camera-Display 1M* with over one million images and 25 camera-display pairs, to train a neural network to learn the representative CDTF. This approach allows us to train the embedding network independently from the representative CDTF. The proposed photographic steganography algorithm learns which features are invariant to CDTF distortion, while simultaneously preserving perceptual quality of the carrier image.

The main contributions in this chapter are: 1) a photographic steganography algorithm based on deep learning architectures; 2) development of a new paradigm for camera-display imaging systems, CDTF-network; 3) Camera-Display 1M: a dataset of 1,000,000 camera-captured images from 25 camera-display pairs.

## 2 Related Work

**Single vs. Dual Channel** Light field messaging, also known as camera-display or screen-camera communication, has been addressed by both the computer vision and the communications literature. Early systems in the communications area concentrate on the screen-camera transfer and do not seek to hide the signal in a display image [48, 78, 8, 45]. In computational photography, single channel systems have been developed for structured light [79] that develop optimal patterns for projector-camera systems.

In the computer vision community, the theme of communicating hidden information in displayed images started with Visual MIMO [80, 64] and continued in other recent work such as InFrame[81, 82, 83, 3] and DisCo [15]. In these dual-channel methods, consistent with our approach, the display conveys information via human observation and the hidden channel transmits independent information via camera-captured video. Prior dual channel methods use fixed filter message embedding using either multiresolution spatial embedding or temporal embedding that requires high frequency displays and high-speed cameras to take advantage of human limitations in perceiving high frequency changes [82, 15, 51].

**Early Steganography** The early work of classic image-processing steganography can be divided into spatial and transform domain techniques. A simple and common form of spatial domain image steganography involves altering the least significant bits (LSBs) of carrier image pixels to encode a message [84]. Small variations in pixel values are difficult to detect visually and can be used to store relatively large amounts of information [85]. In practice, simple LSB steganography is not commonly used because it is easy to detect and requires lossless image compression techniques [86]. More sophisticated LSB methods can be used in conjunction with various image compression techniques such as graphics interchange format (GIF) and JPEG for more complex and difficult to detect steganography [84]. Transform domain techniques of traditional steganography embed using fourier, wavelet, and discrete cosine tranforms [87, 86, 88, 89]. While there is a large body of work in the steganography literature, the methods use fixed filters and these digital methods are not robust to the light transmission in LFM.

**From Fixed Filter to Deep Learning** In recent years, a new class of image steganography algorithms has emerged that utilize deep convolutional neural networks. Pibre [93, 94] and Qian [95] demonstrate that deep learning using jointly learned features and classifiers often outperform more established methods of steganalysis that use hand selected image features. Structured neural learning approaches have been

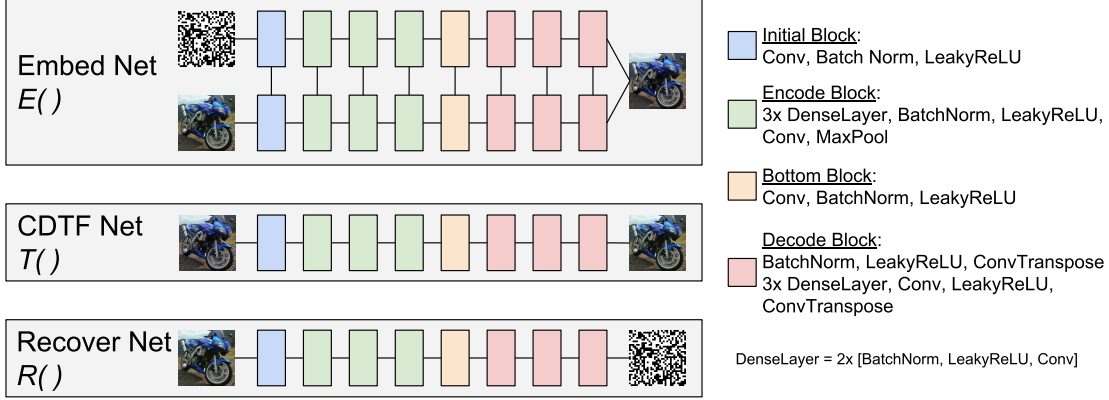


Figure 4.3: Our steganography model’s deep convolutional network architecture.  $R()$  and  $T()$  are both constructed with an identical architecture inspired by U-net for multiscale analysis [90] and Dense blocks for feature reuse [91]. The embedding function  $E()$  combines two images (carrier image and message) into one coded image.  $E()$  has a siamese architecture [92] with separate network halves for carrier image and message. The features for carrier image and message are shared at different scales to ultimately produce a single coded image output. Each half of the siamese architecture of  $E()$  is identical to  $R()$ . Source: [4].

explored that integrate classic image and transform domain steganography techniques, such as LSB selection in a carrier image for a text-based message [96, 97].

For deep steganography, Baluja [1] uses deep feed-forward convolutional neural networks that can directly learn feature representations to embed a message image into a carrier image. Rather than constraining the network to select pixels in a carrier image suitable for embedding, the end-to-end steganography networks are trained with constraints that preserve carrier and message image quality. Hayes devised a similar steganography algorithm based on deep neural networks that utilizes adversarial learning to preserve the quality of the carrier image and limit steganalysis detection [98]. Deep learning approaches such as these have been extended to include video steganography [99], high bits per pixel (BPP) embedding rates [100], resistance to JPEG compression [101], and new deep learning architectures [102, 103]. While our algorithmic approach also uses deep steganography, there is a significant key difference with prior work: we assume our covert message will be electronically displayed, transmitted as light in free space, and then camera-captured. That is, we address the problem of *photographic steganography* for LFM that distinguishes our work from the prior methods

(both classic and deep learning) that address *digital steganography*. Figure 4.2 demonstrates the clear problem in using digital steganography for LFM: the message cannot be retrieved accurately from the camera-captured image.

**Uniqueness of our Approach** Our work is distinct from prior work in that it simultaneously enables: 1) free space light communication, i.e. light field messaging, 2) dual channel communication where the machine-readable message is hidden from the human, 3) deeply learned embedding/recovery, 4) single-frame synchronization-free methodology, and 5) ordinary display hardware with no high frequency requirements.

We are the first to explicitly model and measure the display-camera connection as well as build a first-of-its-kind network and database for learning the coefficients of the camera-display transfer function for use in experiments.

### 3 Methods

We define the terms *message* to refer to the covertly communicated payload, *carrier* to refer to the image used to hide the message, and *coded images* to refer to the combined carrier image and hidden message. Our approach has 3 main components:

- $E()$ : a network that hides a message in a carrier image;
- $R()$ : a network that recovers the message from the coded image;
- $T()$ : a network that simulates the distorting effects of camera-display transfer.

We denote the unaltered carrier image  $\mathbf{i}_c$ , the unaltered message  $\mathbf{i}_m$ , the coded image (carrier image containing the hidden message)  $\mathbf{i}'_c$ , and our recovered message  $\mathbf{i}'_m$ .  $L_c$  and  $L_m$  represent generic norm functions used for image and message loss, respectively. We wish to learn the functions  $E()$  and  $R()$  such that:

$$\begin{aligned}
 & \text{minimize} && L_c(\mathbf{i}'_c - \mathbf{i}_c) + L_m(\mathbf{i}'_m - \mathbf{i}_m) \\
 & \text{subject to} && E(\mathbf{i}_c, \mathbf{i}_m) = \mathbf{i}'_c \\
 & && R(\mathbf{i}'_c) = \mathbf{i}'_m
 \end{aligned} \tag{4.1}$$



In other words, our objective is to simultaneously minimize the distortions to the carrier image and minimize message recovery error. However, this simple formulation will not yield a solution to our problem. A naively trained steganography network will likely learn an embedding function  $E()$  that encodes a message in carrier image LSBs [1]. LSB encoding will be overly distorted by the CDTF, yielding large message recovery errors [2]. Instead, we introduce a third function  $T()$  that simulates CDTF distortion. If  $i_c$  represents an unaltered carrier image, and  $\mathbf{i}'_c$  represents a coded image, let  $\mathbf{i}''_c$  represent a coded image that has passed through the CDTF approximated by  $T()$ , such that  $T(\mathbf{i}'_c) = \mathbf{i}''_c$ . Now we denote a new objective:

$$\begin{aligned}
& \text{minimize} && L_c(\mathbf{i}'_c - \mathbf{i}_c) + L_m(\mathbf{i}'_m - \mathbf{i}_m) \\
& \text{subject to} && E(\mathbf{i}_c, \mathbf{i}_m) = \mathbf{i}'_c \\
& && T(\mathbf{i}'_c) = \mathbf{i}''_c \\
& && R(\mathbf{i}''_c) = \mathbf{i}'_m
\end{aligned} \tag{4.2}$$

The CDTF function  $T()$  must represent both the photometric and radiometric effects of camera-display transfer [2]. This is accomplished by training  $T()$  using a large dataset of images electronically-displayed and then camera-captured using several combinations of cameras and displays. This training procedure is detailed in Section 4. After  $T()$  is trained, the steganography networks  $E()$  and  $R()$  are trained, using  $T()$  as a fixed constraint.

**Network Architecture** Recent trends in deep learning architectures have been to go deeper [104], with more connections between layers [91], and operate at multiple scales [90]. The proposed steganography networks draw heavily from the aforementioned architectures. The 3 networks  $E()$ ,  $R()$ , and  $T()$  all feature dense blocks with feature maps at different scales in the shape of U-Net. Only  $E()$ , the network used for embedding, features a siamese architecture [92]. One half of the network is directly linked to the carrier image  $\mathbf{i}_c$ , while the other half is directly linked to the payload image  $\mathbf{i}_m$ , and produces a single output  $\mathbf{i}'_c$ . The outputs from each pair of blocks are concatenated and passed to subsequent blocks. The network architecture can be seen

in Fig 4.3. See the supplementary material for further details of network architecture such as convolutional layer sizes.

**Perceptual Loss** Broadly, our photographic steganography method has 2 goals: 1) maximize message recovery; and 2) minimize carrier image distortion. For coded image fidelity, our objective function uses the  $L_2$ -norm to measure the difference between  $\mathbf{i}_c$  and  $\mathbf{i}'_c$ . In prior work, photo-realistic image generation using deep neural networks was accomplished with perceptual loss metrics in training [105, 106, 107]. The validity of these perceptual loss metrics have been well established [108]. As is common when training neural networks that produce images as output [109], our perceptual loss metric also includes quality loss. Quality loss is calculated by passing  $\mathbf{i}_c$  and  $\mathbf{i}'_c$  through a trained neural network for object recognition, in this case VGG [110], and minimizing the difference of feature maps at several depths [111].

**Single Frame Advantage** Previous photographic steganography methods such as Visual MIMO [3, 2, 51] and DisCo [15] rely on temporal processing to isolate carrier image content (static) from message content (dynamic). Synchronization issues make this approach difficult in practice. Each display is operating at a frequency independent from each camera and there is no synchronization between camera and display. Even when a camera and display begin in-phase and at complementary frequencies, small changes in operating frequency, lag from computational load, screen-tearing, and rolling-shutter can all cause the system to quickly fall out of sync. The advantage of using a single frame for embedding is that the temporal synchronization problem is avoided.

### 3.1 Camera-Display 1M Dataset

We present *Camera-Display 1M*, a dataset containing over 1 million images collected using 25 camera-display pairs. Images from the MSCOCO 2014 training and validation dataset [112] were displayed on five electronic displays, and then photographed using five digital cameras. The five electronic displays used are the Samsung 2494SJ, Acer

S240ML, Insignia NS-40D40SNA14, Acer Predator XB271HU, and Dell 1707FPt. The five cameras used are the Pixel 2 smartphone, Basler acA2040-90uc, Logitech c920 webcam, iPhone 8 smartphone, and Basler acA1300-30uc. The chosen hardware represents a spectrum of common cameras and displays. To achieve a set of 1M images, 120,000 images of MSCOCO were chosen at random. Each camera-captured image is cropped, warped to frontal view, and aligned with its original. The measurement process was semi-automated and required software control of all cameras and displays. The time-consuming acquisition process has produced a comprehensive dataset that will be made publicly available [75] along with the trained CDTF network parameters. See Figure 4.4 for examples of how different hardware in the imaging pipeline significantly alters the appearance of the same images.

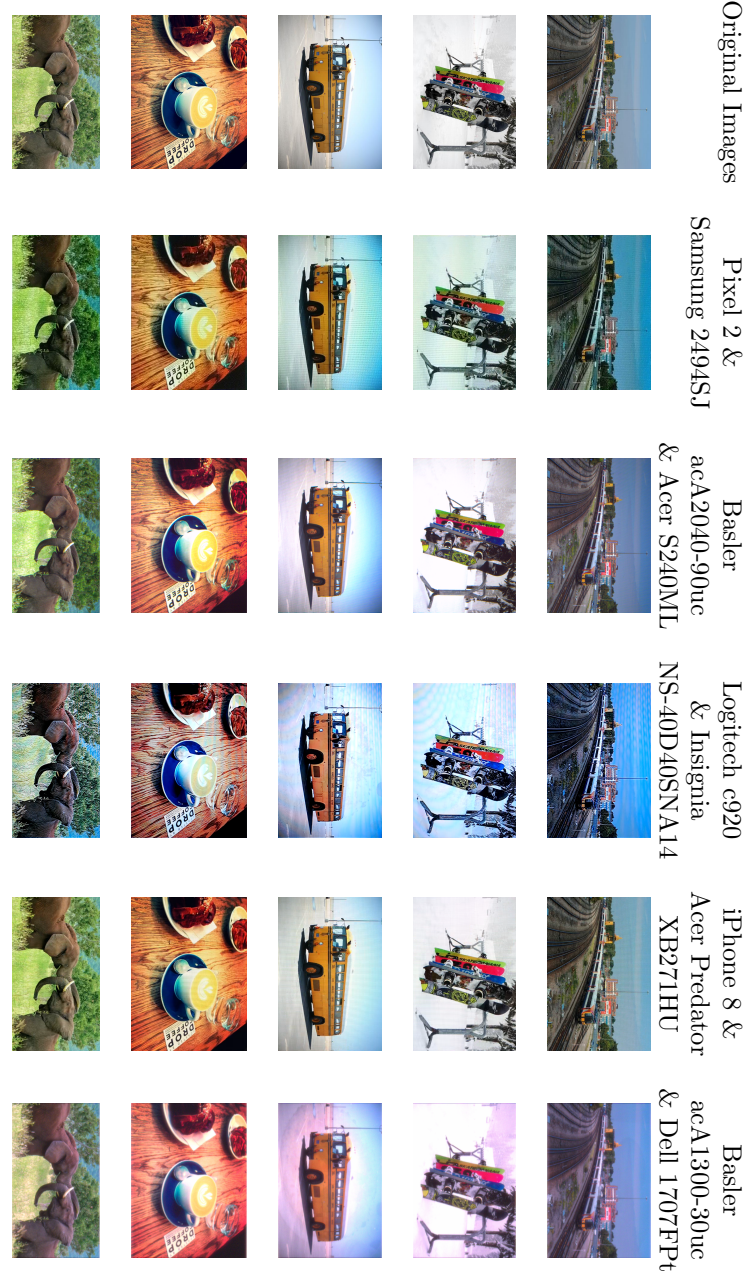


Figure 4.4: *Camera-Display 1M* examples: Our dataset contains over 1 million images collected from 25 camera-display pairs. Each column corresponds to a different camera-display pair (5 of 25 are shown). Camera properties (spectral sensitivity, radiometric function, spatial sensor pattern) and display properties (spatial emitter pattern, spectral emittance function) cause the same image to appear significantly different when displayed and captured using different camera-display hardware. Source: [4]. (Best viewed as zoomed-in PDF.)

### 3.2 Training $T()$

The network  $T()$  is trained using 1,000,000 image pairs,  $\mathbf{i}_{\text{COCO}}$  representing the original image and  $\mathbf{i}_{\text{CDTF}}$  representing the same image displayed and camera-captured. These images used for training are MS-COCO images [112] that are rendered on an electronic display and then camera-captured using 25 camera-display pairs. The objective of  $T()$  is to simulate CDTF distortion by outputting  $\mathbf{i}_{\text{CDTF}}$  given  $\mathbf{i}_{\text{COCO}}$  as input. The objective function we wish to minimize is:

$$T_{\text{loss}} = \|\mathbf{i}_{\text{COCO}} - \mathbf{i}_{\text{CDTF}}\|_2 + \lambda_T * \|\mathbf{VGG}(\mathbf{i}_{\text{COCO}}) - \mathbf{VGG}(\mathbf{i}_{\text{CDTF}})\|_1. \quad (4.3)$$

We include a perceptual loss regularizer for  $T()$  to preserve the visual quality of the network output  $\mathbf{i}'_{\text{c}}$ . The perceptual loss weight  $\lambda_T$  is 0.001.  $T()$  is trained for 2 epochs using the Adam optimizer with a learning rate of 0.001,  $\beta = (0.9, 0.999)$ , and no weight decay [113]. Total training time was 7 days.

### 3.3 Training $E()$ and $R()$

The networks  $E()$  and  $R()$  are trained simultaneously using 123,287 images from MS-COCO [112] for  $\mathbf{i}_{\text{c}}$ , and 123,287 messages for  $\mathbf{i}_{\text{m}}$ . The objective of  $E()$  is to produce a coded image  $\mathbf{i}'_{\text{c}}$  that is visually similar to  $\mathbf{i}_{\text{c}}$ , and encodes all the information from  $\mathbf{i}_{\text{m}}$  such that it is robust to CDTF distortion. The objective of  $R()$  is to recover all information in  $\mathbf{i}_{\text{m}}$  despite CDTF distortion. When training  $E()$  and  $R()$  with  $T()$ , our goal is to satisfy Equation 4.2.  $T()$  is pretrained and placed in the training loop for  $E()$  and  $R()$ . The output of  $E()$ , passes through  $T()$  before becoming the input to  $R()$ . As  $E()$  and  $R()$  are trained and updated through backpropagation, the pretrained  $T()$  network remains static. The objective functions we wish to minimize are:

$$E_{\text{loss}} = \|\mathbf{i}_{\text{c}} - \mathbf{i}'_{\text{c}}\|_2 + \lambda_E * \|\mathbf{VGG}(\mathbf{i}_{\text{c}}) - \mathbf{VGG}(\mathbf{i}'_{\text{c}})\|_1. \quad (4.4)$$

$$R_{\text{loss}} = \phi * \|\mathbf{i}_{\text{m}} - \mathbf{i}'_{\text{m}}\|_1$$

Again here, we include a perceptual loss regularizer for  $E()$  to preserve the visual quality of the network output  $\mathbf{i}'_{\mathbf{c}}$ . The perceptual loss weight  $\lambda_E$  is 0.001, and the message weight  $\phi = 128$ .  $E()$  and  $R()$  are trained for 3 epochs using the Adam optimizer with a learning rate of 0.001,  $\beta = (0.9, 0.999)$ , and no weight decay [113, 114]. Total training time was 18 hours. The networks  $E()$ ,  $R()$ , and  $T()$  were all trained using PyTorch 0.3.0 with an Nvidia Titan X (Maxwell) compute card [115].

## 4 Experiments and Results

To study the efficacy of our approach, we constructed a benchmark with 1000 images, 1000 messages, and 5 camera-display pairs. The images are from the MSCOCO 2014 test dataset, and each message contained 1024 bits. Two videos were generated, each containing 1000 coded images embedded using a trained LFM network, one trained with  $T()$  and one without. As shown in Table 4.1, the proposed LFM algorithm trained with  $T()$  achieved 7.3737% BER, or 92.6263% correctly recovered bits on average for frontally photographed displays. The same algorithm achieved 14.0809% BER when camera and display were aligned at a 45 deg angle. The example in Figure 4.5 illustrates the differences between coded images  $\mathbf{i}'_{\mathbf{c}}$  generated with and without the CDTF network  $T()$  in the training pipeline. All BER results in this chapter are generated without any error correcting codes or radiometric calibration between cameras and displays.

	Pixel 2 & Samsung 2494SJ	Basler acA2040-90uc & Acer S240ML	Logitech c920 & Insignia NS-40D40SNA14	iPhone 8 & Acer Predator XB271HU	Basler acA1300-30uc & Dell 1707FPt
LFM without $T()$ , frontal	49.961%	50.138%	50.047%	50.108%	50.042%
LFM with $T()$ , 45° (ours)	29.807%	15.229%	<b>10.217%</b>	5.1415%	10.01%
LFM with $T()$ , frontal (ours)	<b>10.051%</b>	<b>6.5809%</b>	10.333%	<b>5.0732%</b>	<b>4.8305%</b>

Table 4.1: BER for various camera-display pairs (*lower is better*). One thousand randomly generated  $32 \times 32$  (1024-bit) messages were embedded into one thousand previously unused MSCOCO images. Message recovery was evaluated using 5 cameras and 5 displays. The distances between camera and display range from 23cm to 4.3 meters. The table shows the mean BER for each camera-display pair. While 0% BER would be a perfectly recovered message, 50% BER corresponds to randomly classified bits. Each device was operated with its default manufacturer settings for normal use.

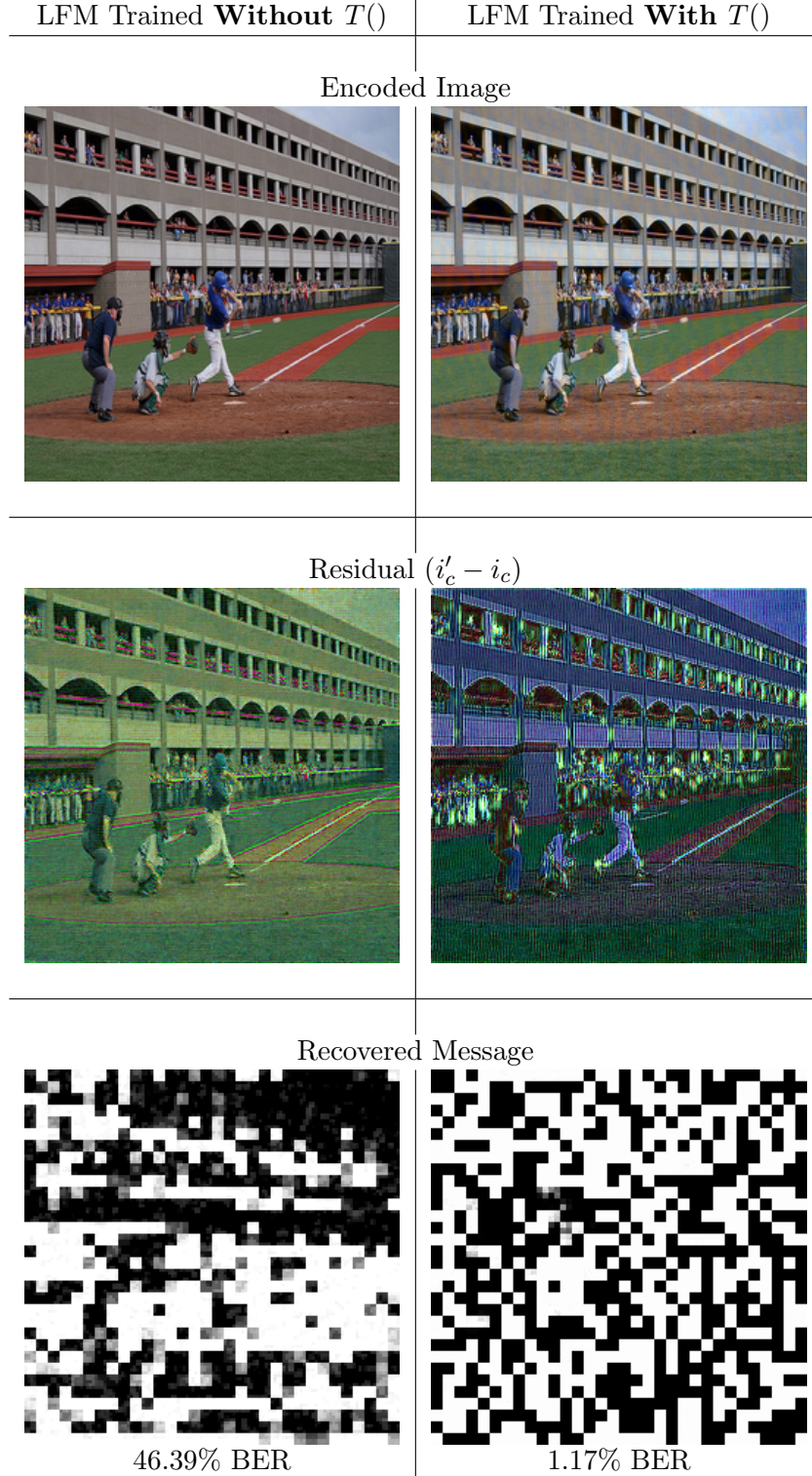


Figure 4.5: Coded images generated using the same carrier image and message, produced with two otherwise identical steganography architectures: **Left:** trained without the CDTF; **Right:** trained with  $T()$  to model CDTF. The per-pixel changes ( $i_c - i'_c$ ) in the two middle images are multiplied  $\times 50$  for visibility. Notice the significant changes to coded image appearance that our photographic steganography model learns that anticipate the CDTF (right). This experiment was performed using the Pixel 2 camera and Acer Predator XB271HU display. Source: [4].



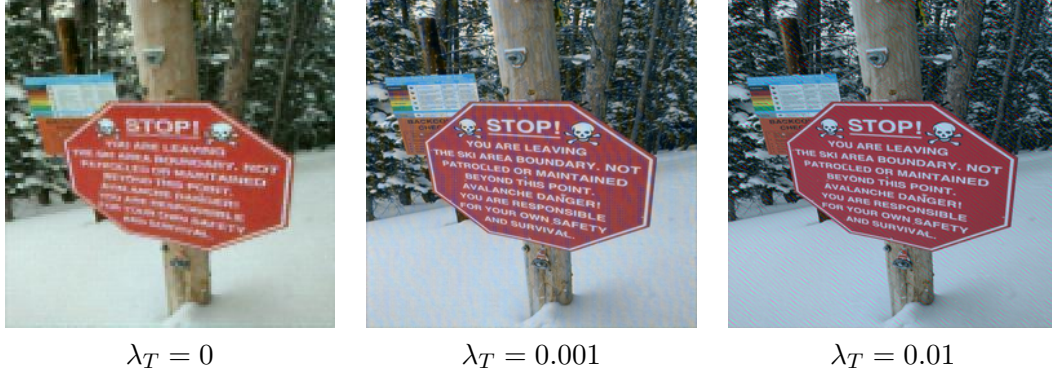


Figure 4.6: Examples of coded images generated by our photographic steganography model with various perceptual loss weights in training. As the perceptual quality metric  $\lambda_T$  is increased, the image becomes sharper and has fewer color shift errors. If  $\lambda_T$  is too large, BER increases, as is the case when  $\lambda_T = 0.01$ . Source: [4]. (Best viewed as zoomed-in PDF)

We wish to understand the effects of perceptual loss in our steganography model. In particular, we examine the effects of  $\lambda_T$  by varying its weight in the loss function during training. Figure 4.6 features an ablation study of the effects of perceptual loss. Figure 4.7 features an example of the same image and message camera-captured at different angles. The LFM algorithm trained without  $T()$  is analogous to digital steganography deep learning techniques, and was unable to successfully recover coded messages even when frontally viewed, the simplest case. Figure 4.5 illustrates the difference that the inclusion of  $T()$  in LFM training makes. Without  $T()$ , the message is encoded as small per-pixel changes that are near-uniform across the image. With  $T()$ , the message is encoded as patches where the magnitude of pixel changes varies spatially. We show an empirical sensitivity analysis of camera exposure settings in Figure 4.8. Our LFM method is robust to overexposure and underexposure, provided pixels are not in saturation.

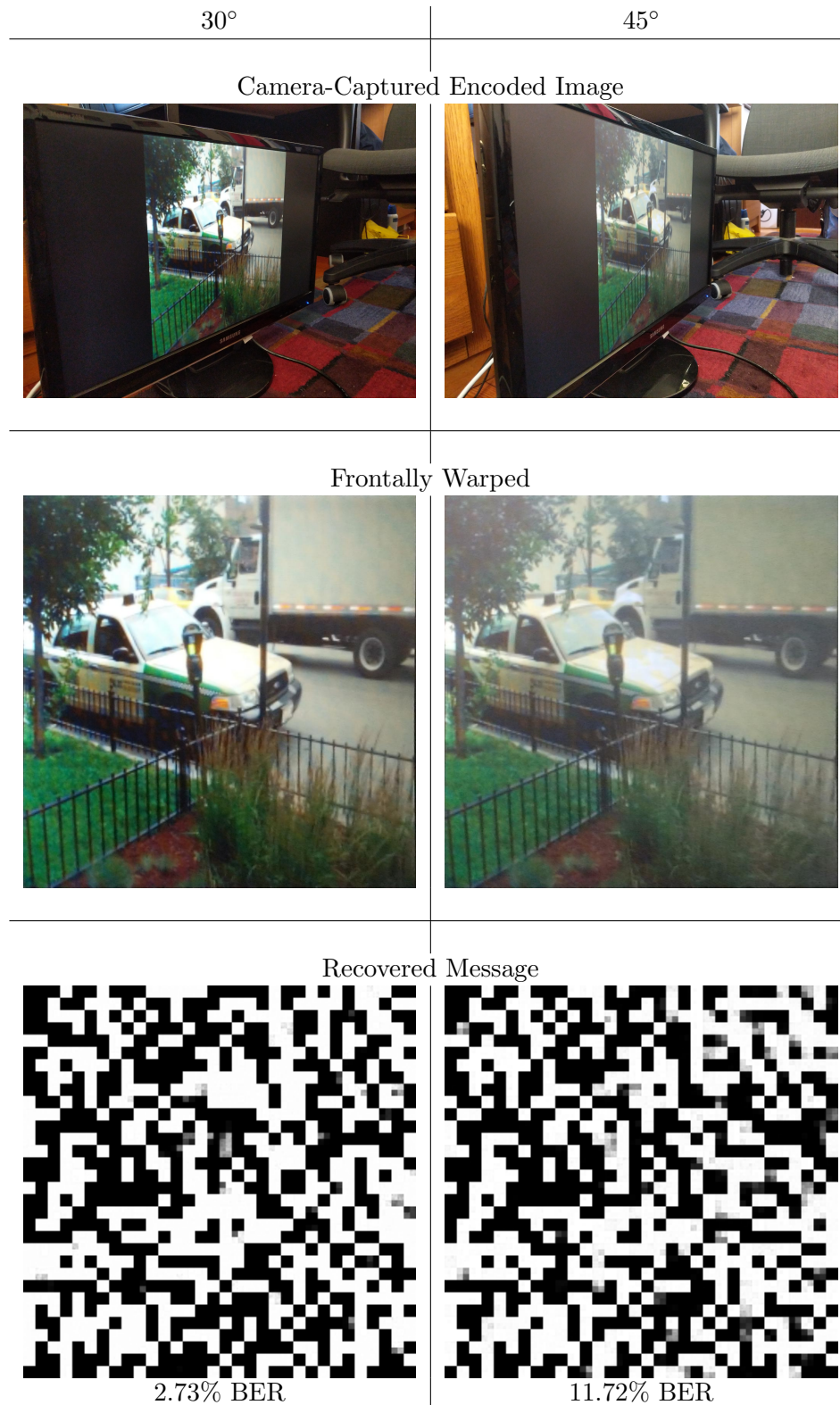


Figure 4.7: Camera display angle has a significant effect on message recovery. This experiment was performed using the Pixel 2 camera and Samsung 2494SJ display. Our LFM method performs well for oblique views, but experiences a steep dropoff in BER as the camera-display angle increases. Source: [4].

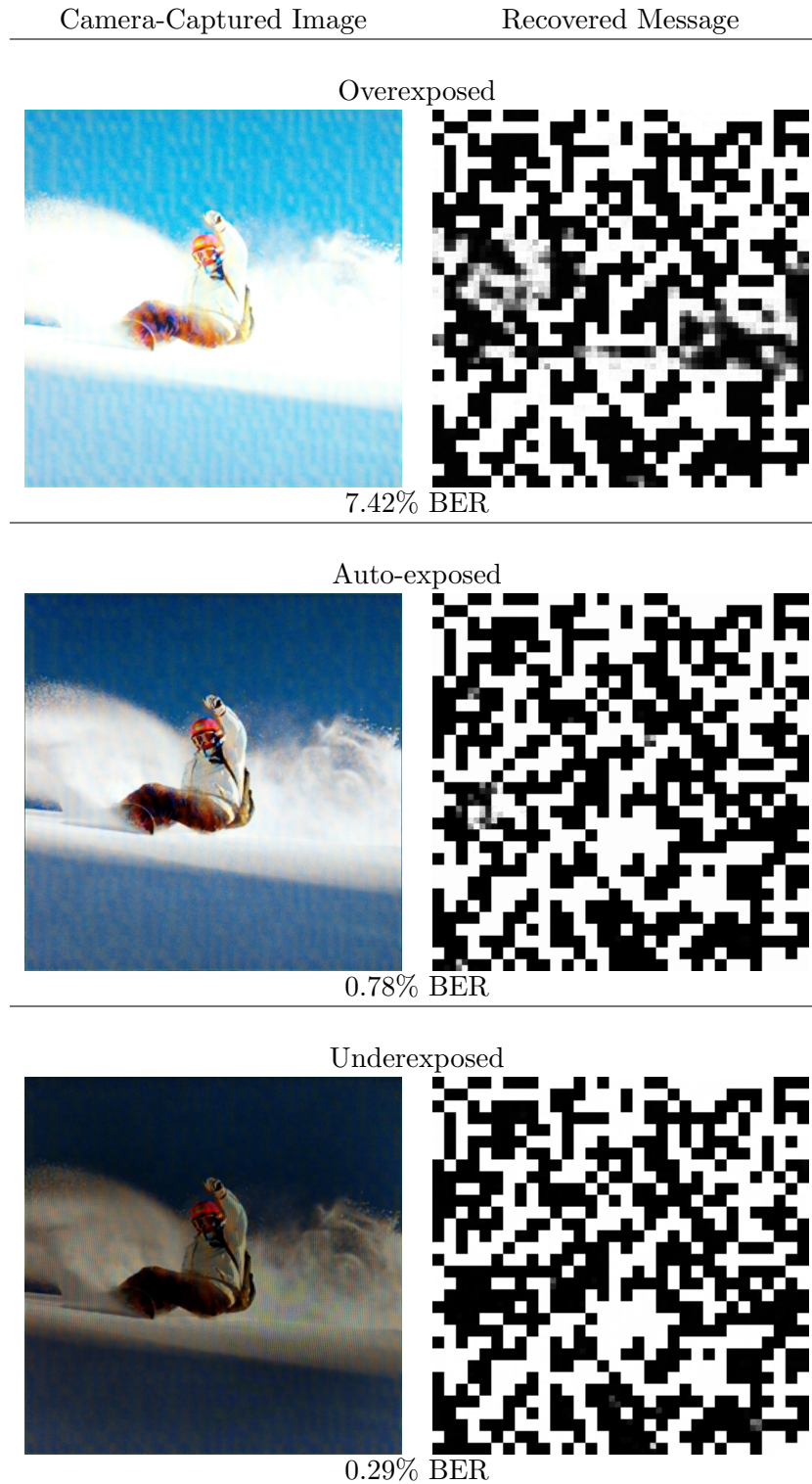


Figure 4.8: Our approach is robust to modifications of camera exposure, yielding low BER for multiple settings. Underexposure performs better than overexposure because the message cannot be recovered from the saturated snow pixels in the overexposed image. This experiment was performed using the Pixel 2 camera and Acer Predator XB271HU display. Source: [4].

Finally, we motivate the need for photographic steganography with a comparison to existing methods. Are existing synchronization-free steganography algorithms such as Baluja [1] sufficient for photographic message transfer? As shown in Figure 4.2, even simple binary messages are not stably transmitted photographically using existing methods. Our CDTF simulation function  $T()$  is trained with 25 camera-display pairs, but we want to know how well  $T()$  generalizes to new camera-display pairs. Using the 1000-image, 1024-bit test dataset, we test two additional cameras and two additional displays. We create coded images using various embedding algorithms and measure message recovery accuracy for each of the four camera-display pairs. Table 4.2 shows that LFM trained with  $T()$  significantly outperforms existing methods, even when camera and display are at a  $45^\circ$  angle.

## 5 Discussion

In this chapter, we extend deep learning methods for digital steganography into the *photographic* domain for LFM where coded images are transmitted through light, allowing users to scan televisions and electronic signage with their cameras without an internet connection. This process of *photographic steganography* is more difficult than digital steganography because radiometric effects from the camera-display transfer function (CDTF) drastically alter image appearance [2]. We jointly model these effects as a camera-display transfer function (CDTF) trained with over one million images. The resulting system provided embedded messages that are not detectable to the eye and recoverable with high accuracy.

Our LFM algorithm significantly outperforms existing deep-learning and fixed-filter steganography approaches, yielding the best BER scores for every camera-display combination tested. Our approach is robust to camera exposure settings and camera-display angle, with LFM at  $45^\circ$  outperforming all other methods at  $0^\circ$  camera-display viewing angles. Along with our LFM algorithm, we introduce Camera-Display 1M, a dataset of 1,000,000 image pairs generated with 25 camera-display pairs. Our contributions open up exciting avenues for new applications and learning-based approaches to photographic steganography.

	Sony Cybershot DSC-RX100 & Lenovo Thinkpad X1 Carbon 3444-CUU	Sony Cybershot DSC-RX100 & Apple MacBook Pro 13-inch, Early 2011	Nikon Coolpix S6000 & Lenovo Thinkpad X1 Carbon 3444-CUU	Nikon Coolpix S6000 & Apple MacBook Pro 13-inch, Early 2011
DCT [116], frontal	50.01%	50.127%	50.001%	49.949%
Baluja [1], frontal	40.372%	37.152%	48.497%	48.827%
LFM without $T()$ , frontal	50.059%	49.948%	50.0005%	49.997%
LFM with $T()$ , 45° (ours)	12.974%	15.591%	27.434%	25.811%
LFM with $T()$ , frontal (ours)	<b>9.1688%</b>	<b>7.313%</b>	<b>20.454%</b>	<b>17.555%</b>

Table 4.2: Generalization to new camera-display pairs: Our LFM model generalizes to new camera and display hardware, outperforming traditional fixed-filter Discrete Cosine Transform (DCT) [116] and deep-learning-based [1] steganography approaches. Here, we show BER for 1000 1024-bit messages transmitted with 4 new camera-display pairs that were not in the training set.

## Chapter 5

# Deep CNNs as a Method to Classify Rotating Objects based on Monostatic Radar Cross Section

### 1 Introduction

When illuminated with a narrowband radar signal, an object reflects incident energy and the reflectance depends on the object's geometry and material properties. The amount of energy that is reflected directly back toward the source of illumination is a function of its monostatic RCS (Radar Cross Section). As an object changes orientation, the RCS changes as well. We wish to classify the 3D shape of objects based only on a time series of monostatic RCS as the object moves according to force-free rigid body motion. Our set of target objects includes right circular cones, right circular cylinders, rectangular planes, spheroids, and trapezoidal prisms. The target object set varies in size with respect to a geometric parameter for each class (e.g. radius and height variation for cylinders). The chosen geometric properties in the test set are selected by radar wavelength so that each object is modeled as a Perfect Electrical Conductor (PEC). Labelled data, i.e. RCS of known objects, are required to train and test our supervised classifier. We create a large dataset of geometric objects and their corresponding RCS time-series signals.

To simulate real-world conditions, the input signals for testing are corrupted by Gaussian noise and *Swerling dropout*. The Swerling Model [117] is a standard method for determining the detectability of an object based on SNR and waveform characteristics. The instantaneous probability of detecting each object at a given time is explicitly included in order to make the performance closer to real world operation. If the Signal to Noise Ratio (SNR) at a given time point is too small, a real-world radar system may be unable to separate the object from noise and will therefore be unable to detect the

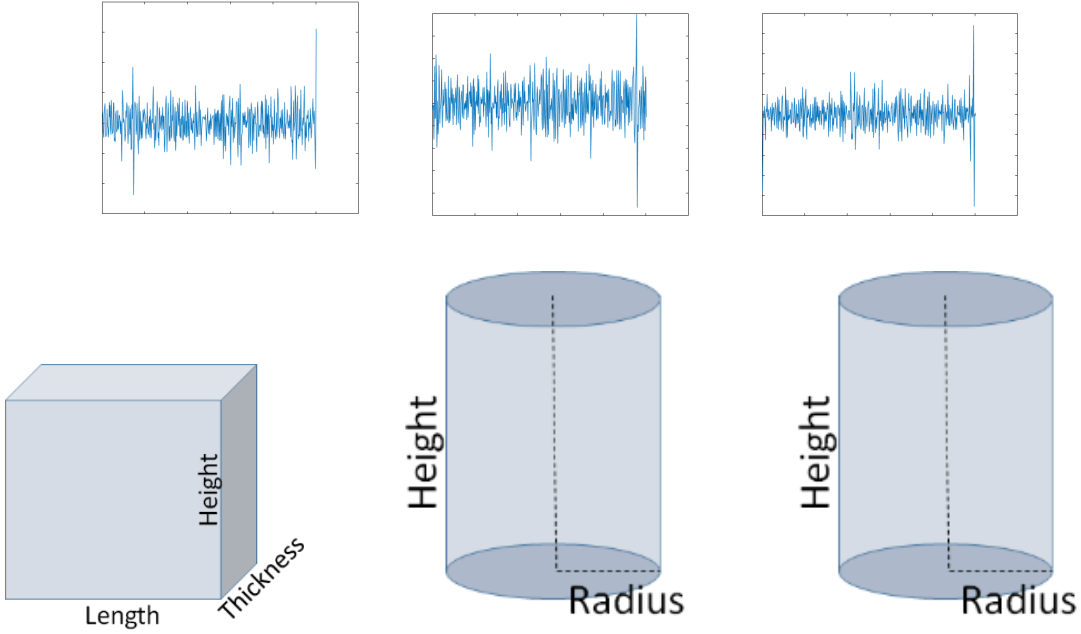


Figure 5.1: Our goal is to correctly predict object shape family from a noisy monostatic RCS signal. RCS is highly sensitive to motion, and the rotation rates and viewing angles are unknown to the classifier. For example, objects may be rotating very fast or very slow about multiple axes. These signals contain added white Gaussian noise and a Swerling detection model, where the probability of detection is smaller for lower RCS values results in missing data points. A convolutional neural network (CNN) is used to *learn* the separating features that accurately recognize each object class overcoming the challenge of noisy data, missing data and unknown trajectories. Source: [5].

object and estimate its RCS.

A subset of the generated signals are used to train a feed-forward convolutional neural network classifier. We employ an end-to-end learning architecture, where signal features and the classifier are jointly solved for. The inputs are a series of RCS samples over time as the object rotates through free space. These objects belong to one of four shape families, illustrated in Figure 5.2. When the rotation is simple and follows a known path (as shown in Figure 5.3, top row), the problem is trivial. However, the problem becomes substantially more difficult when the motion parameters are unknown (see Figure 5.3, bottom row). Examples of the difficulty of the goal are illustrated in Figure 5.1.

In this work, we successfully classify the shape family for rotating objects with unknown roll rates, tumble rates, and unknown initial orientations. We train deep



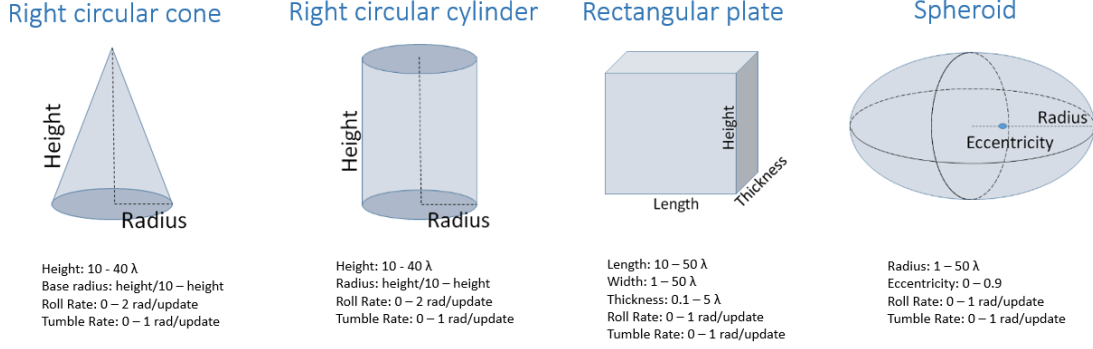


Figure 5.2: The four shape families correspond to four target classes in our classifier. Each shape class has a range of geometric parameters and motion parameters. The parameter ranges are listed under each shape.  $\lambda$  is wavelength of the incident radar signal. Source: [5].

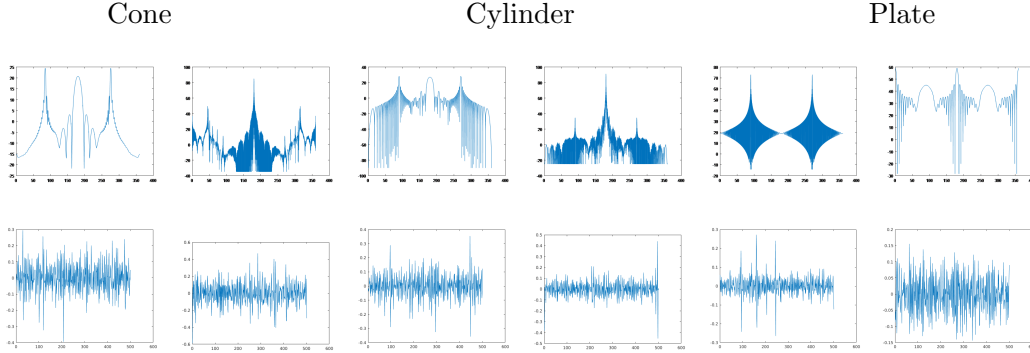


Figure 5.3: There is tremendous variation among the cone, cylinder, and plate RCS signals on the top row. Those signals have rotation about a fixed axis at a relatively slow speed and zero noise. The bottom row features 2 more realistic cone, cylinder, and plate RCS signals. The salient features present in the top examples are now gone. Source: [5].

neural network classifiers that return the probability of each signal belonging to each shape family. The deep learning training and testing is implemented using PyTorch, a machine learning and optimization library for the Python programming language [115]. The SVM and Decision Tree algorithms are implemented using the SciPy library for the Python programming language [118]. To our knowledge, our methods are the first application of deep learning for object shape classification using monostatic radar signals.



## 2 Related Work

Producing an accurate representation of a target object’s narrowband monostatic RCS is a challenging problem. Radar specific properties such as wavelength and sampling rate, as well as object-specific properties such as surface material, shape, and motion may dramatically influence the resulting RCS time series. In this application, the objects under investigation are geometrically simple, convex shapes with uniform material construction. The incident energy wave is assumed to be a simple plane wave. The environment is not modeled, except for the addition of Gaussian noise. Due to these constraints, the physical optics (PO) approximation is appropriate to produce realistic returns. Open source RCS signal generation tools such as the Matlab toolbox PO-Facets are readily available [119] and have been used to approximate RCS of aircraft models [120].

A powerful new class of supervised machine learning algorithms called *convolutional neural networks* (CNNs) leverage optimization to learn complex latent features for robust classification. This family of algorithms is called *deep learning* when networks contain many convolutional layers. In 2012, a convolutional neural network significantly outperformed all other algorithms on the object classification dataset ImageNet [121] and CNNs have become the algorithm of choice for image recognition in computer vision [110, 122, 123, 104, 124].

Traditional neural networks have been used for radar classification tasks for decades, often derived from architectures developed for speech recognition such as the time-delay neural network [125, 126]. Early work on neural networks for processing radar signals were applied to identifying the number and type of radar emitters in a simulated multisource environment [127]. Pulse-train radar signal classification and source identification remains a topic of active research [128, 129]. Another recent challenge for neural networks in radar is the identification of radar jamming signals [130, 131]. Traditional neural networks have been applied to: SAR imagery for ground terrain classification [132] and crop classification [133]; microwave radar for classifying pedestrians

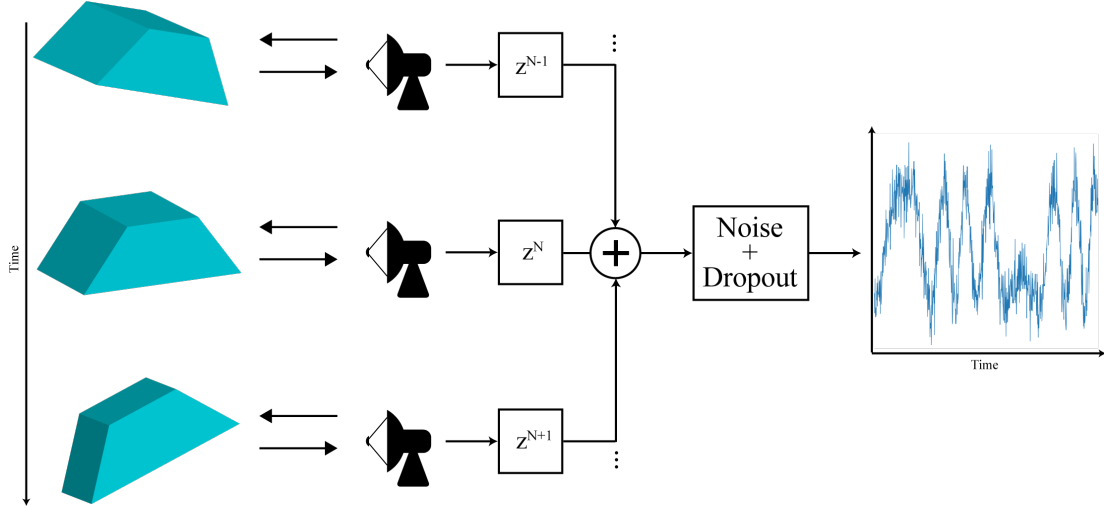


Figure 5.4: The POFacets library was used to generate RCS signals from geometric shape models. Generalized Euler motion, additive Gaussian noise, and Swerling 2 dropout are then incorporated to generate the final signal. Source: [5].

and vehicles [134]; doppler radar for identify human breathing [135]; ground penetrating radar for the classification of geological structures [136]; forward scattering radar for identifying very small marine targets [137].

While traditional neural networks have been used widely in radar classification tasks, modern deep learning and CNNs are beginning to take hold in recent applications [138, 139, 140, 141, 142]. The success of the 2D CNNs on standard color images has translated well into radar applications. While most deep learning networks are designed for 2D imagery and can be directly applied to radar-based imagery, however, the RCS time series signals in our work are one-dimensional signals. In fields such as natural language processing [143] and medical applications [144], 1D CNNs have provided successful classification. In this work, we leverage successful deep networks for 2D image recognition, but adapt the networks to the 1D monostatic RCS signals.

Multi-static radar systems utilize a set of receivers and transmitters to create multiple 1D RCS signals of a target object. In prior work, multistatic RCS signals are classified individually using CNNs [138, 145] and the average of multiple CNNs [146] for multistatic contextual target signatures. The monostatic system addressed in our work contains a single collocated receiver-transmitter pair, compared to multistatic systems which have one or more spatially separated receivers and transmitters. The

classification problem of monostatic RCS signals is particularly challenging since the signals do not contain contextual information from multiple sources.

### 3 Generating RCS Signals

The first step in RCS classification is generating 3D models of our target objects. The parameters of these objects are listed in Figure 5.2. 128 geometric models were generated, each corresponding to one of four shape classes in the primary experiments. For each of the 3D models, POFacets is used to generate narrowband monostatic RCS values. In the case of monostatic radar, we assume that the radar source and receiver are at the same location. The radar frequency is kept constant. It is important to note that in the physical optics model, RCS behavior depends only on the size of the object in wavelengths. Thus we can arbitrarily set the chosen frequency to  $0.3GHz$  while preserving the general behavior of any wavelength. Since the 3D model parameters are scaled by wavelength, this allowed for unit shape size parameters. POFacets is used to generate narrowband monostatic RCS responses, sensitive to object rotation parameterized by  $\theta$  and  $\phi$ . The mapping is done by specifying an angular sweep from  $0^\circ$  to  $180^\circ$  at high sampling intervals of  $0.1^\circ$ . Symmetry about the shapes allows us to simulate to a maximum rotation of  $180^\circ$ .

#### 3.1 Generalized Euler Motion

Once an RCS map had been generated, a motion path is drawn over the surface and the map is be interpolated. The target objects are assigned tumble, roll, and initial rotation angle. The initial conditions are then propagated following the physics of rigid body motion in the presence of no external forces (free motion). A quaternion model is used to generate the motion path parameterized by  $\theta$  and  $\phi$  over the precomputed 2D RCS map. The roll and tumble parameters are bound by the values described in Figure 5.2. For each shape class, the center of mass and moment of inertia are calculated and used for the simulation of realistic, geometry-dependent object motion.

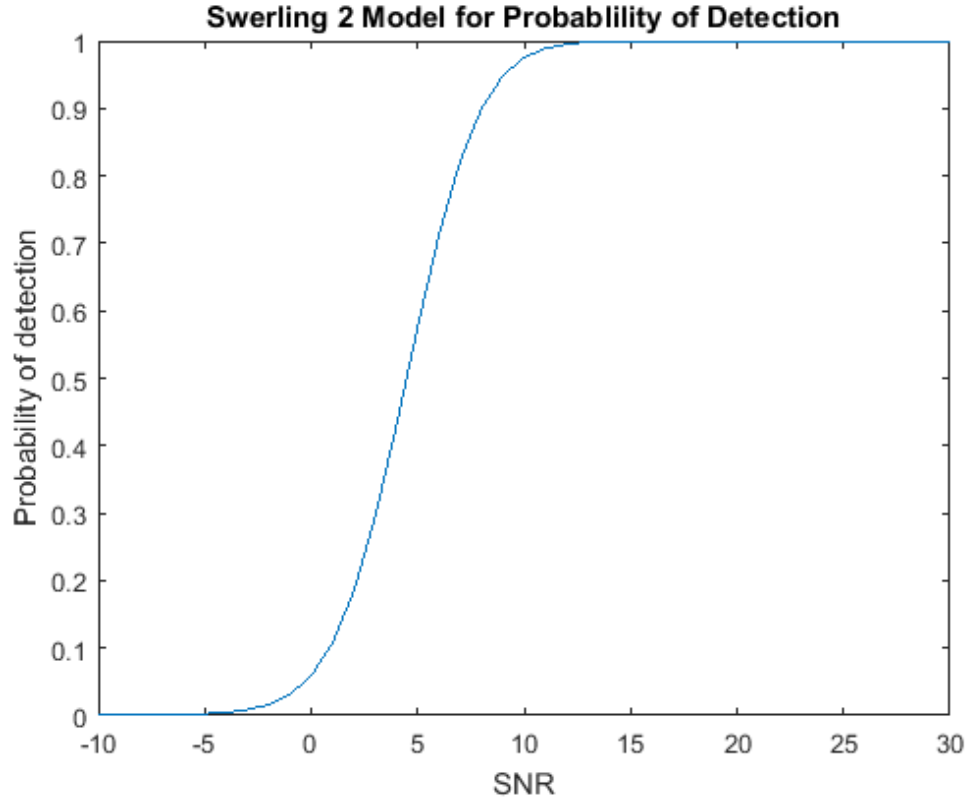


Figure 5.5: Swerling detectability is an important parameter in our model. As the RCS SNR decreased, so does the probability of detection. According to the above graph, SNRs of  $25dB$  and  $15dB$  provide almost no dropped measurements. But for  $SNR = 5dB$ , the probability of detection drops significantly, to roughly 50%. The RCS measurements with the lowest magnitude have a greater likelihood of being dropped to 0. Although Swerling dropout did have a major effect on our results, it often preserves larger RCS values in the time series signal, and the larger RCS values are expected to play a more substantial role in feature selection. Source: [5].

### 3.2 Randomizations in Motion Parameters

It would be relatively easy to classify RCS signals from objects at integer-valued roll, tumble, and viewing angle. To make the problem more realistic and challenging, randomizations were applied to the values of each parameter. A random variable  $x$  with  $\mu = 1$  and  $\sigma = 0.5$  was multiplied with the viewing angle ( $\theta$  and  $\phi$ ), tumble rate, and rotation rate for each signal. The random variation allows for the construction of a database where the same 2D RCS map could be used to generate multiple signals. The ability to scale motion parameters with random jitter allowed the creation a nearly equal number of signals between the four classes, even though there were more

Parameters	A4	B4
Number of classes	4	4
Tumble rate (rad/sec*max)	0.015, 0.1, 0.5	0.015, 0.1, 0.5, <b>1</b>
Roll rate (rad/sec*max)	0.015, 0.1, 0.5	0.015, 0.1, 0.5, <b>1</b>
Signal to noise ratio (dB)	25, 15	25, 15, <b>5</b>
Viewing vector angle (degrees)	0, 20	0, 20, <b>40, 60</b>
Swerling model	2	2
Probability of false alarm	0.0001	0.0001
Number of pulses	10	10
Signal length (samples)	501	501
Number of signals	121,320	363,960

Table 5.1: Generation parameters for A4 and B4 datasets

3D models created for plates. CNN performance is generally improved when there are equal number of training examples in all classes.

### 3.3 Update Rate, Swerling, Gaussian Noise, Gradients, and Pyramids

A realistic radar model has a finite update rate. The number of samples as an object rotates are related to the update rate (in Hz) and the rotation rates (in radians/second). In this study the kinematic bounds of the objects are defined in radians/update, thus the performance of a highly sampled signal that rotates quickly is the same as as if it were rotating more slowly with a corresponding decrease in radar update rate. The motion parameters are specified in radians per update. The radar update rate is arbitrarily set to 1 Hz. To simulate realistic distortions of each RCS value, Gaussian noise and a Swerling detectability model are incorporated into each RCS signal. The addition of Gaussian noise transforms the RCS from a truth value to an estimate. The specific parameters can be found in Table 5.1.

To summarize, the objects under test have complex motion with tumble, roll, and variable viewing angles, yielding complex time series of RCS estimates. The signals are noisy and have missing data points. Each RCS signal dataset contains variable values for each of the aforementioned parameters. Therefore, the same classifier is expected to correctly label RCS signals from objects moving at highly varied speeds in highly varied motion paths with different amounts of noise.

## 4 Experiments

Two datasets are created using the methods described. One is used for training, and the other for evaluation/testing. The parameters used to create these dataset are listed in Table 5.1. The datasets in this chapter are named A4 and B4 respectively because they both contain four classes but have different parameter values.

All experiments were run on a Ubuntu 16.04 machine with 32GB of RAM, a Xeon E5-1620 v4 @ 3.5GHz x 8 CPU, a Samsung 860 EVO SSD, and a Nvidia Titan X (Maxwell edition) GPU. The PyTorch and SciPy library versions used for training and evaluation are 0.1 and 1.1 respectively.

### 4.1 Residual Network

Our 1D residual network architecture is inspired from He et al. [104]. Two-dimensional  $3 \times 3$  convolutional filters were replaced by one-dimensional  $3 \times 1$ ,  $5 \times 1$ , and  $7 \times 1$  filters, but the original block module structure and skip connections are maintained. See Figure 5.6 for a detailed view of the 18-layer network architecture. The residual network was run over 30 epochs and updated using the Adam [114] optimizer with a learning rate of 0.001. Unlike the original implementation of ResNet, batch normalization is done during training to avoid overfitting. The batch size for training is 128 signals for all models except for the 152-layer residual network due to GPU memory constraints and is instead run with a batch size of 32 signals. The learning rate is decayed by 70% if the current validation accuracy does not improve compared to the average of the previous five validation accuracies. The network with the lowest validation error is saved and used to evaluate the test data. The 18-layer residual network requires five minutes to train while the 152-layer residual network requires nearly three hours to train. The time required to evaluate a signal with the listed hardware is on the order of tens of microseconds, allowing real time signal classification.

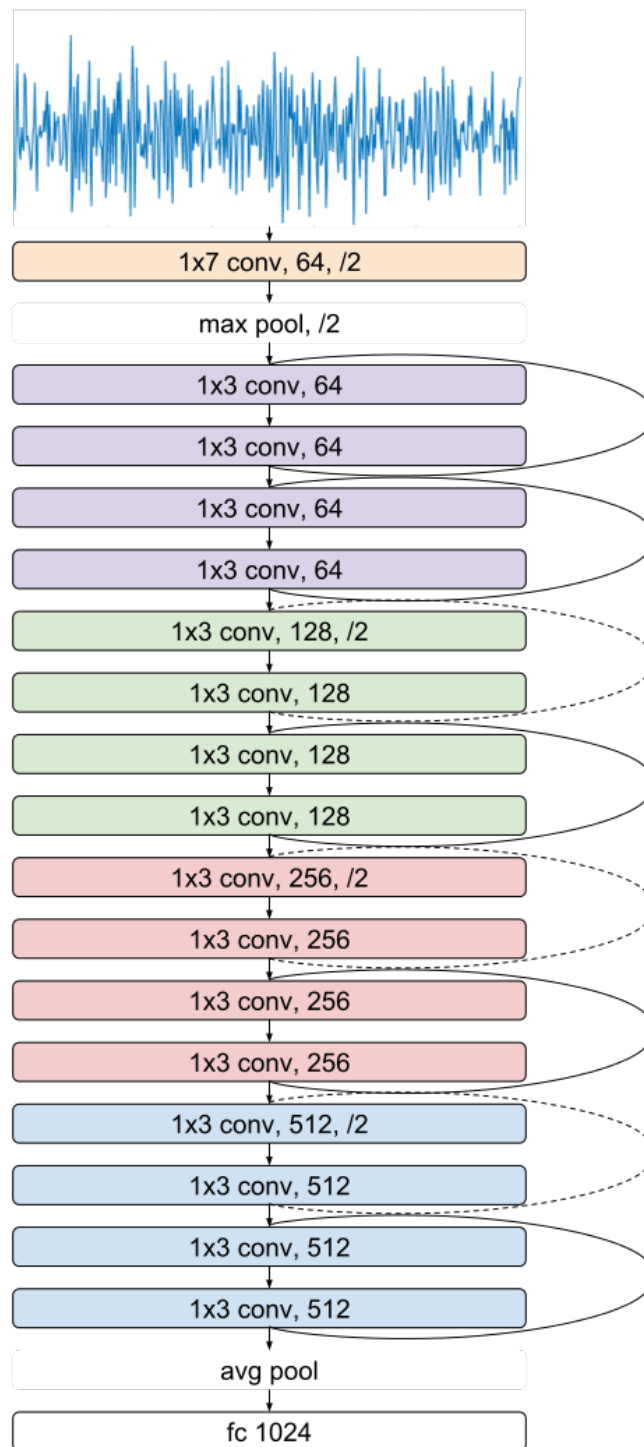


Figure 5.6: An 18-layer convolutional network is trained to analyze a noisy RCS signal. The architecture is strongly inspired by ResNet [104]. Skip connections are shown as curved arrows. Unlike ResNet, batch normalization is incorporated into the model. Source: [5].

	Cone	Cylinder	Plate	Sphere	Trapezoidal Prism
Train (#)	24,201	25,189	29,041	19,311	2,258
Test (#)	1,871	2,038	2,214	1,623	254
Train (%)	24.2	25.2	29.0	19.3	2.3
Test (%)	23.4	25.4	27.3	20.3	3.2
Models	11	12	30	5	1

Table 5.2: The number of each respective model in the A5 dataset.

## 4.2 Expanding the A4 Dataset

In secondary tests we expand our four class dataset to include a new trapezoidal prism class. We augment the dataset to answer the question of how our model performance would be affected by the addition of a smaller class of signals. This object is selected such that it closely resembles one of the original classes, i.e. the plate class. One trapezoidal prism class model was created. The new dataset distribution is recorded in Table 5.2. The number of signals for the new class is significantly lower than the other classes. We call this dataset A5 because it contains the same motion parameters as A4 but has an extra shape class.

## 4.3 Siamese Network

Our initial hypothesis was that our residual network would misclassify signals belonging to the class with the fewest instances, confusing them with one of the larger classes. If we assume one class will be confused, the loss function will be minimized by misclassifying signals in the smallest class. In order to test our hypothesis, we compare the performance of the residual network with a siamese network. A siamese network consists of two feature extractor modules, each outputting a lower dimensional, compared to the original input, feature vector. The goal of our siamese network is to cluster signals from the same class in close proximity while moving signals from different classes farther apart in feature space. This network is chosen such that the smaller class is less likely to be grouped with another class. The feature extractor modules share the same parameter so that the output vectors can be compared symmetrically. The 18-layer residual networks are used as the feature extractors in the siamese architecture. As with our other trained CNNs, the siamese network is trained using the Adam optimizer



with batch sizes of 128 signals for 30 epochs. The learning rate was also initialized and adjusted congruently. The comparator or loss function requires a margin hyperparameter to separate signals of different classes:

$$L = \sum_i^N y^i \cdot \|x_1^i - x_2^i\|_2^2 + (1 - y^i) \cdot \max(0, m - \|x_1^i - x_2^i\|_2^2) \quad (5.1)$$

The loss function encourages signals in feature space synthesized from the same type of model to converge while forcing signals in feature space belonging to different models farther apart. A CNN generates a fixed length feature representation of the input signal from learned feature extractors. The similarity between feature representations of two signals,  $x_1$  and  $x_2$ , is measured with the  $L2$  distance metric. The binary label  $y = 1$  if the signals are from the same shape primitive model then, and  $y = 0$  if the signals are not from the same primitive. Signals from the same shape primitives are forced closer in feature space. Whereas, signals from different shape primitives are forced apart if the distance between the feature representations are closer than the margin  $m$ . Since the network requires two signals, evaluation is computed by measuring the similarity between a test signal and a set of signals from the training dataset. Several methods were attempted as classifiers but ultimately a nearest neighbor classifier performed with the greatest accuracy. An input signal first passes through the feature extractor network to produce the corresponding test signal feature vector. The test feature vector is compared to a set of training feature vectors. The most similar feature vector to the test feature vector assigns its label to the test vector. Other methods such as a  $k$ -nearest neighbor with  $k > 1$  and a support vector machine (SVM) were also used but did not perform as well.

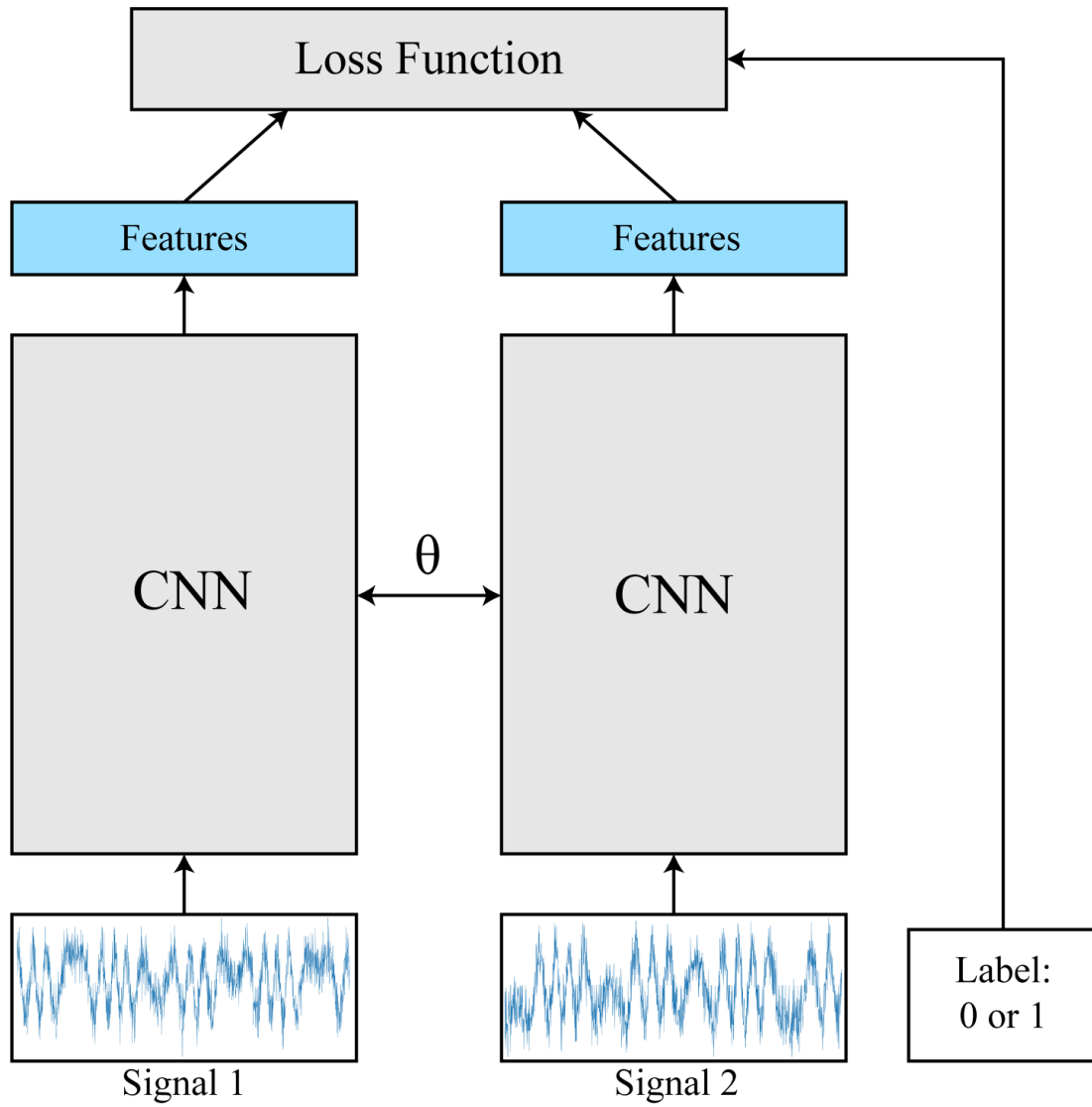


Figure 5.7: Two signals are fed into two CNNs with shared parameters. The output feature vectors are compared via the Siamese network loss function 5.1. The target label is equal to one if the two signals belong to the same class and zero otherwise. Source: [5].

#### 4.4 Robustness Test

In order for the classifier to be utilized in real-world applications, it must make accurate predictions on signals with previously unseen distortions. Signal distortions such as occlusion, saturation, and clutter can affect monostatic RCS signals. Occlusion, in this work, is defined as zeroing a subset of a signal’s RCS values. Clutter is defined as random amplitude spikes at random locations within a signal. Saturation or clipping is a hard cutoff at a set threshold that limits a signal’s amplitude. Subsampling is the removal of a random contiguous section of a signal. Occlusion differs from subsampling because occluded signals have the same number of samples after the distortion is applied unlike signal subsampling. As a robustness test, the network is trained on dataset A4 which only contains signals distorted by noise and Swerling dropout. The trained network then evaluates a test set of the A4 dataset that is distorted by one of the previously mentioned distortions. The degree of distortion is varied in each test, e.g. the test signals are saturated to 75% of their maximum amplitude. The residual architecture can receive signals of various dimensions as its input because of an average pooling layer before the end of the feature extractor module. Subsampling is implemented by circularly shifting the signal by a random integer and then setting the last  $n$  elements to zero.

#### 4.5 Refiner Network

This section is inspired by the work done by Shrivastava et al. [147], where the authors train a refiner network to make generated images appear more realistic. This network resembles a generative adversarial network (GAN) [122] where a generating network tries to create “realistic” data and a discriminator network decides whether the data is real or fake. The generator network iteratively improves the generated image while the discriminator network learns to more accurately discern the real and fake data apart. Instead of generating data from a noise distribution, as with the classic GAN example, a refiner network converts simulated data into data that more resembles the realistic data. In this work we use a refiner network to make our simulated RCS signals look

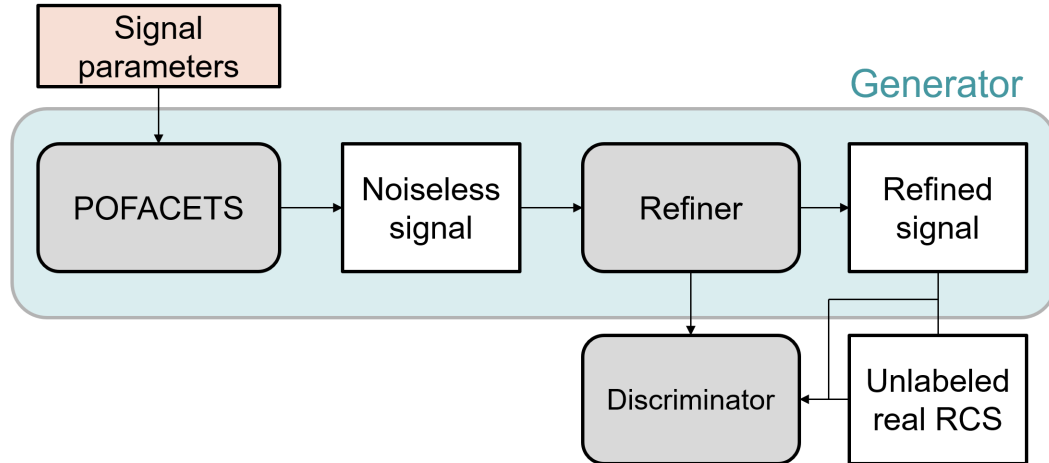


Figure 5.8: The refiner network and the discriminator work in a similar adversarial manner as a generative adversarial network. The refiner optimizes the simulated signals to look more like the unlabeled realistic data while the discriminator tries to distinguish the difference between the refined and realistic signals. Source: [5].

like simulated signals with added noise. The refiner network maintains the structure of our signal while adding features to make it appear more like the signals with noise. The parameters used for the simulated dataset are similar to A4 dataset except that no noise is added to the signal and rotation and roll rates are decreased.

The refiner network is a 3-layer CNN that takes a simulated signal as input and outputs a refined signal of the same size. The discriminator network is a 5-layer CNN that receives the refined signal as input and outputs a vector probability map. The probability map determines which parts of the input signal appear realistic to the discriminator. The refiner and discriminator networks have separate loss functions and are trained iteratively. The refiner network's loss function is a combination of the distance between the input signal and the generated signal and the likelihood that the discriminator believes that the refined signal is real. The discriminator network's loss function is a combination of the likelihood that the discriminator believes that the refined signal is real and the likelihood that the discriminator is unsure that the real data is real. Both networks are trained for 50 epochs with the Adam optimizer. For each epoch the refiner network is trained twice while the discriminator is only trained once.

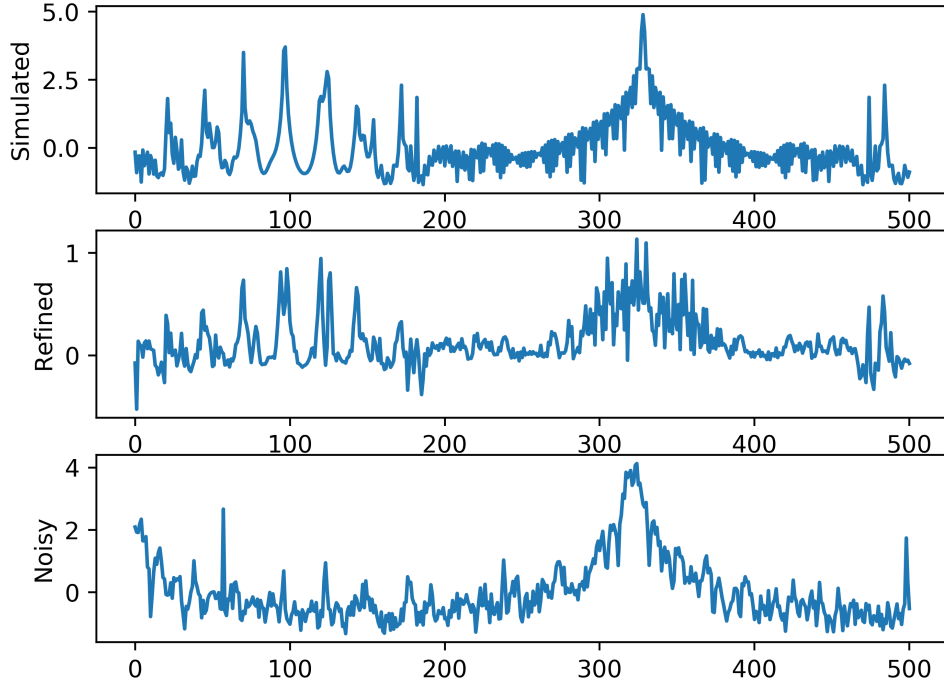


Figure 5.9: The result of the refiner network is shown above. The refiner network takes the signal in (a) as input and returns the signal in (b). It learns to make this transformation by observing signals with noise like the signal in (c). Source: [5].

## 5 Results

In this section we explore the performance of our trained CNNs on our generated datasets. We also compare different architecture performance using an augmented dataset, investigate the robustness of our classifier, and explore improving our simulated data post-generation.

### 5.1 Classification on A4 and B4 Datasets

Several residual networks with layer depths shown in Figure 5.10 are trained as described in the experiments section, on both A4 and B4 datasets. Best performance is achieved using the 152-layer residual networks, with classification error scores of 2.5% and 2.0% on datasets A4 and B4 respectively, as shown in Figure 5.10. While the general trend implies that deeper networks perform better, this is not always true. The

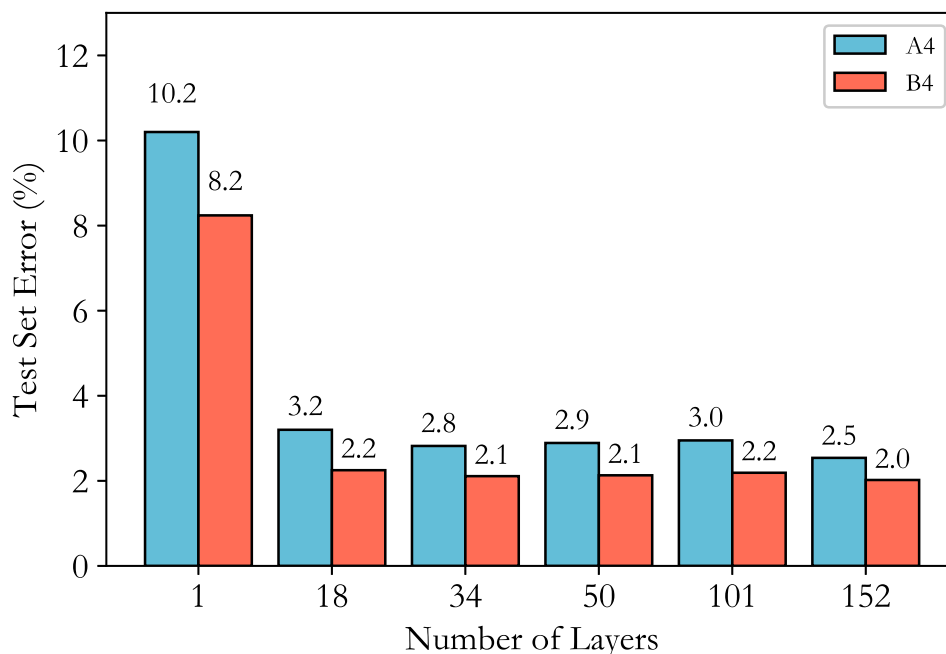


Figure 5.10: Several residual networks of different lengths are evaluated on both the A4 and B4 datasets. As the number of layers in the architecture increases the test error on either dataset decreases but only achieves marginal improvement past a depth of 18 layers. Source: [5].

101-layer network performs slightly worse than the 50-layer and 152-layer networks for both the A4 dataset (2.9% vs. 3.0% vs 2.5% for A4) and the B4 dataset (2.1% vs. 2.2% vs. 2.0%). Since all of these networks were trained with the same data, hyperparameters, and appropriately scaled architecture for the given depths, it is difficult to explain this fluctuation in test performance. Test performance saturates for the 18-layer network, and performance changes only slightly for larger networks. As network size increases, so does the ability to learn more complex features. But larger networks also have a propensity to overfit if the dataset used for training is not sufficiently large and representative of the distribution of each class. When overfitting occurs, training accuracy will continue to improve while test accuracy continues to degrade. Since Figure 5.10 features test error, and the A4 and B4 datasets are sufficiently large, the networks are likely not overfit, but at saturation for test accuracy given the complexity of useful signal features. Likely, the small deviations in test performance stem from

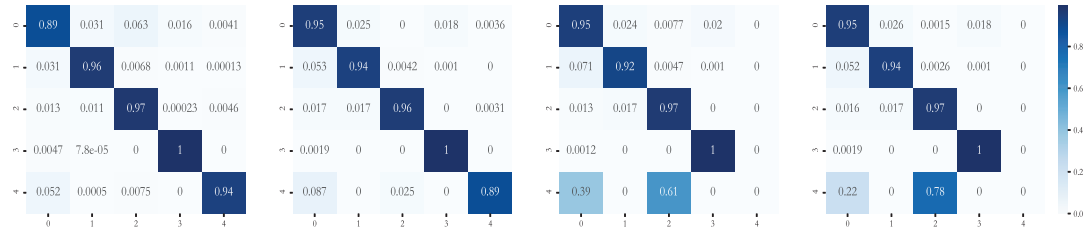


Figure 5.11: The confusion matrices for all siamese networks and the single residual network. Confusion matrices starting from the left to the right belong to the single network, the siamese network with nearest neighbor, siamese network with k nearest neighbor, and siamese network with support vector machine. The classes are enumerated as (0) cone, (1) cylinder, (2) plate, (3) spheroid, and (4) trapezoidal prism. Source: [5].

each network converging on different local minima in the optimization plane. Initial conditions and when training is stopped may have effects on which minima a network is likely to converge on.

Models trained on the B4 dataset perform better than models trained on the A4 dataset across all network depths. As a baseline, a neural network and non-residual convolutional neural network were trained and evaluated on the A4 dataset with the corresponding test errors, 29.5% and 6.1%. The neural network contains six layers, dropout, and non-linear layers. Increasing the number of layers in the neural network did not significantly improve results. The non-residual convolutional neural network contained 18 layers and is trained with the same training parameters described in the experiments section. When the number of layers in the non-residual convolutional network was increased, performance plateaued and then began to degrade.

Our classification results for the residual networks may appear counter-intuitive at first glance, since CNNs typically perform worse on datasets that have more variation. Datasets with more variation are simply more difficult to learn because the CNN will have to learn specific filters to deal with that variation. Not only does the B4 dataset contain more signals but it contains faster roll and tumble rates. The faster roll and tumble rates for our signals actually increases the amount of information per sample because the models we use to generate our signals have large distinct edges and smooth

	SVM	DT	1L-CNN	RN18	RN152
Original	0.557	0.485	0.898	0.968	0.975
SS	0.863	0.849	-	-	-
TR	0.707	0.607	-	-	-
TR+SS	0.742	0.828	-	-	-

Table 5.3: Accuracy performance of support vector machine (SVM), decision tree (DT), single-layer convolutional neural network (CNN), and residual network (RN) algorithms on the A4 dataset. The leftmost column represents the signal features that were used by each classification algorithm. Common signal statistics (SS) represents feature vectors comprised of the mean, standard deviation, and extremum of a signal. Transform representations (TR) represent feature vectors comprised of coefficients from the Fourier and Wavelet transforms of a signal. Since convolutional neural networks learn a feature representation, only the original signals are used as input.

surfaces. If instead the models used had rough surfaces and less distinct edges, information would be lost by increasing the roll and tumble rates. The B4 dataset also contains signals with lower SNR rates and more varied viewing angles, which decrease the amount of information within the signals. Regardless of the size of the network, test performance on the B4 dataset was greater than on the A4 dataset. It was for this reason that the A4 dataset was selected to create new datasets and to further train/test our models. If a more difficult dataset is used, then there will be a clearer distinction between the results of more advanced networks.

In addition to neural networks, we assess the performance of other machine learning classification algorithms such as support vector machines (SVM) and decision trees (DT) on the A4 dataset. The SVM algorithm utilizes the radial basis function kernel with a gamma value equal to reciprocal of the number of input features. Multiple one-against-one classifiers are aggregated to form the final SVM classifier. As for the DT, the Gini criterion is used to measure the quality of the split in the tree and decision nodes are randomly chosen to be further split. The minimum number of samples to be a leaf node is set to five, and the minimum number of samples required to split a decision node is two. Unlike deep learning algorithms, features must be manually crafted for the SVM and DT classifiers to attain optimal performance. For comparison the SVM and DT classifiers are trained and evaluated on the complete length signals from the A4 dataset and achieve accuracies of 55.7% and 48.5% respectively, as shown in Table 5.3.



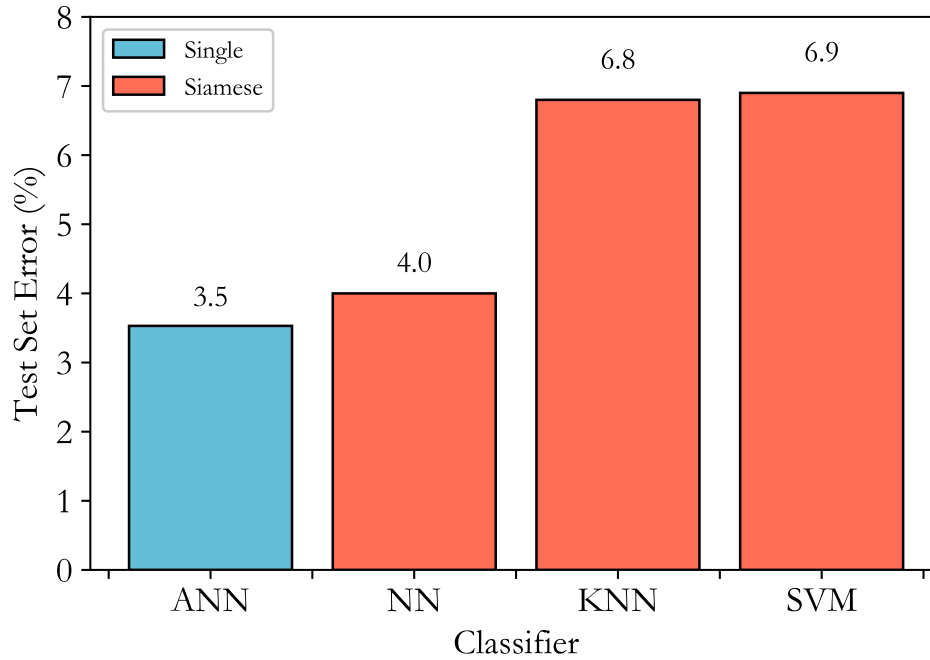


Figure 5.12: A single residual network's performance on the A5 test dataset is compared to the performance of three Siamese networks with various classification layers. ANN stands for artificial neural network, NN is nearest neighbor, kNN is k nearest neighbor, and SVM stands for support vector machine. Source: [5].

Common signal statistics (SS) such as minimum and maximum are combined with low order cumulants [148] to form a representation of the RCS signals. This representation improves upon the previous the accuracy of the classifiers to 86.3% and 84.9%. Following the work of Byl [149] and Zhang [150], more complex descriptive features such as Fourier Transform frequency responses and Wavelet Transform coefficients are used to represent the signals. Specifically, the Fast Fourier Transform generates frequency coefficients and the Discrete Wavelet Transform (DWT) symmetrically pads signals during the transform in order to avoid inaccurate calculation of the DWT. The first 50 coefficients from each transform are concatenated to form the feature vector representation. This method, which we call transform representations (TR), is combined with the SS features to achieve accuracies of 74.2% and 82.8%. For reference our one layer CNN (1L-CNN) has a test accuracy of 89.8% on the A4 dataset, Figure 5.10.

## 5.2 Classification on A5 Dataset

The siamese network structure has been used on a variety of tasks such as signature matching and facial identification with high performance [151, 152]. This type of network performs most effectively when the number of classes in a dataset is large and the number of data per class is relatively low. The architecture’s unique comparator function forces input from the same class to cluster in high dimensional space and input from different classes to be farther apart in high dimensional space. The loss function for a typical CNN classifier is the negative log likelihood function which does not contain any constraint on how far apart the output vectors of the feature extractor module are. The A5 dataset contains the same set of parameters as A4 but includes an additional geometric model of trapezoidal prism. The additional class contains only one model and makes up a small portion of the total signals in the A5 dataset.

The A5 dataset is a superset of the A4 dataset, but augmented with an additional and easily-confused shape class. The results of this experiment are shown in Table 5.4. The single residual network outperforms all types of the siamese networks in terms of overall accuracy as shown in Figure 5.12. Initially it appears that the lack of clustering term in the objective function does not reduce performance on the A5 dataset, however the CNN could maintain high accuracy even while misclassifying all of the signals in the newest class. To further investigate this result the precision, recall, and the F1-score of each class is calculated and shown in Table 5.4. The siamese networks with the k-nearest neighbor and support vector machine classifiers misclassified the trapezoidal prism class in every case. The single residual network and the siamese network with the nearest neighbor classifier were both able to correctly classify the trapezoidal prism class a majority of the time.

In Table 5.4 we can see that the F1-score for the trapezoidal class is greater in the single network section than the siamese network section. Overall the average F1-score across classes is 0.948 and 0.956 for the single network and siamese network respectively. If we weigh the F1-score by the number of signals per class there is an even larger difference in performance. The weighted F1-score of the single network and siamese

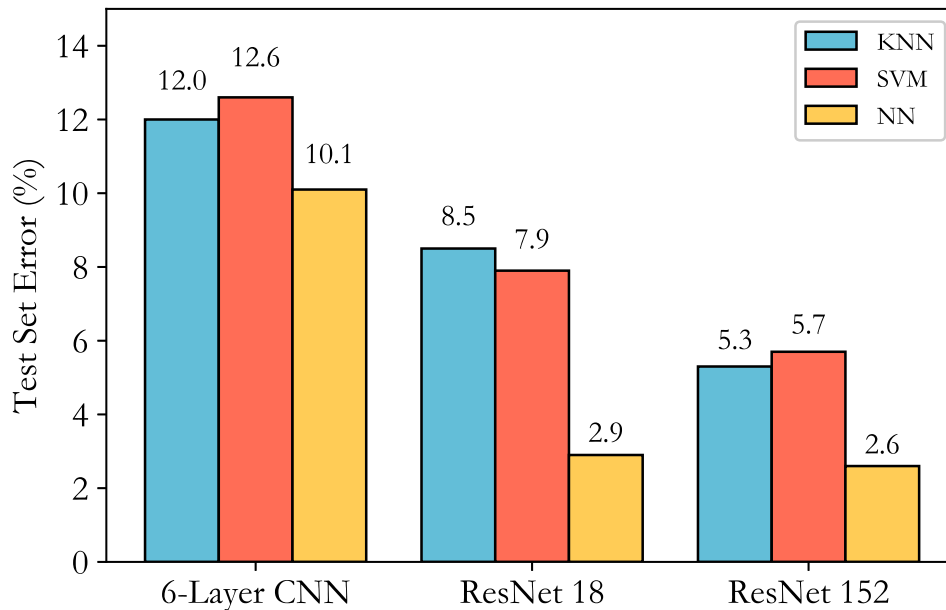


Figure 5.13: Three different classifier modules are compared after a CNN feature extractor of varied depths. The nearest neighbor classifier achieves the highest overall accuracy consistently across all architectures tested. Source: [5].

network are 0.947 and 0.959 respectively. It appears that the single network showed high performance on the trapezoidal prism class because it misclassified more of the signals in the cone class. The siamese network with the nearest neighbor classifier performs well because the feature extractor module is better able to separate the clusters for each class. Intuitively we expect the k nearest neighbor and support vector machine classifiers to outperform the nearest neighbor classifier, but our results in Figure 5.13 suggest otherwise. The dimensionality of the output vector from the feature extractor module may be a potential reason that the nearest neighbor classifier performs better. As the number of dimensions increase, the k nearest neighbor algorithm tends to perform worse due to the increasing space in between points.

### 5.3 Robustness Metric Performance

A CNN classifier's ability to handle noisy input data can be evaluated in multiple ways, such as testing on a novel set of data with distortions seen in the training data or testing

Class	Single Network			Siamese Network+NN		
	Precision	Recall	F1-Score	Precision	Recall	F1-Score
Cone	0.94	0.89	0.91	0.92	0.95	0.94
Cylinder	0.96	0.96	0.96	0.95	0.94	0.95
Plate	0.94	0.97	0.95	0.99	0.96	0.98
Spheroid	0.98	1.00	0.99	0.98	1.00	0.99
Trapezoidal Prism	0.93	0.94	0.93	0.95	0.89	0.92

Table 5.4: Accuracy performance comparison between a single residual network and a siamese network with a NN classifier on the A5 dataset

on a novel set of data with distortions unseen in the training data. Monostatic radar signals can have a variety of distortions in real applications such as signal occlusion, clutter, sensor saturation, subsampling, or a combination of several. Since generating a dataset with every combination of signal distortions is unwieldy, we instead decide to evaluate our system’s robustness to distortions by evaluating our model on data with distortions not seen in the training data. The results shown in Figure 5.14 are the F1-score per class from a single 18-layer residual network. However, the robustness results for networks with more layers is nearly identical and not presented. The evaluation set was generated via the method described in the experiments section.

The network performs remarkably well on signals that have been occluded by even 75% of the total signal, even though no dropout layers are used to train the model. Occlusion may not affect our network significantly because the rotation rates used in our dataset generation are relatively large and occasionally the shape model is rotating several times within the full window of sampling. Even if the signal is occluded significantly, some signals with high rotation rates may contain enough information for classification. However signals generated with slower rotation rate parameters do not appear to complete rotations multiple times within a full window. For these cases the CNN is able to discern the object within a limited viewing window. The CNN is however very sensitive to signal clutter, accuracy-per-class drops as soon as clutter is introduced. Clutter in this work is the addition of random peaks in a signal and CNNs are sensitive to slight distortions to input data. This distortion is similar to the distortion created by adversarial attacks such as FGSM [153], except that we are adding distortions with random amplitudes at random locations. Most CNNs are not robust

to adversarial attacks and it appears that clutter approximates an adversarial attack in this domain. The CNN is resilient to signal saturation up to roughly 15%, then performance decreases significantly soon after. Signals with heavy saturation begin to appear indistinguishable from each other, and the filters that the CNN uses to detect features cannot distinguish between each class. The rise in F1-score of some of the classes seems to be an artifact of the dataset instead of a feature of the network. The final distortion is subsampling the input signal. This measure is similar to the occlusion distortion but the number of total samples in the signal do not change in the occlusion distortion. The results of subsampling show that the CNN can use signals with lengths as small as 25 samples as input and achieve a reasonable F1-score. The performance halves when input size is 5% of its original length. The siamese network evaluated with the robustness metric is not included because the previously mentioned siamese testing method compares an input signal to a subset of the training data. Since the training data does not contain the distortions of the evaluation data, unsurprisingly, the siamese network performs very poorly.

#### 5.4 Classification on Refined Dataset

In order to compare the difference between the simulated dataset and the refined dataset we train separate three layered convolutional neural networks. The network’s performance was evaluated by classifying simulated signals with added white Gaussian noise. The simulated signals with added noise were also used as “real” data in the refiner network training. Overall the model’s performance on the evaluation dataset is greater when the model is trained using the refined dataset by 3.5%. The accuracy of the network trained on the simulated subset A4 dataset is 86.7%, while the accuracy of the network trained on the refined dataset was 90.2%.

No simulator can perfectly model the all of the nuances and variables that are required to create real data. Therefore training a CNN on simulated data typically does not perform well on real data. This does not mean that networks should be trained with only real data because representative real data is difficult and expensive to obtain. Real data is also potentially biased in terms of only representing certain occurrences and

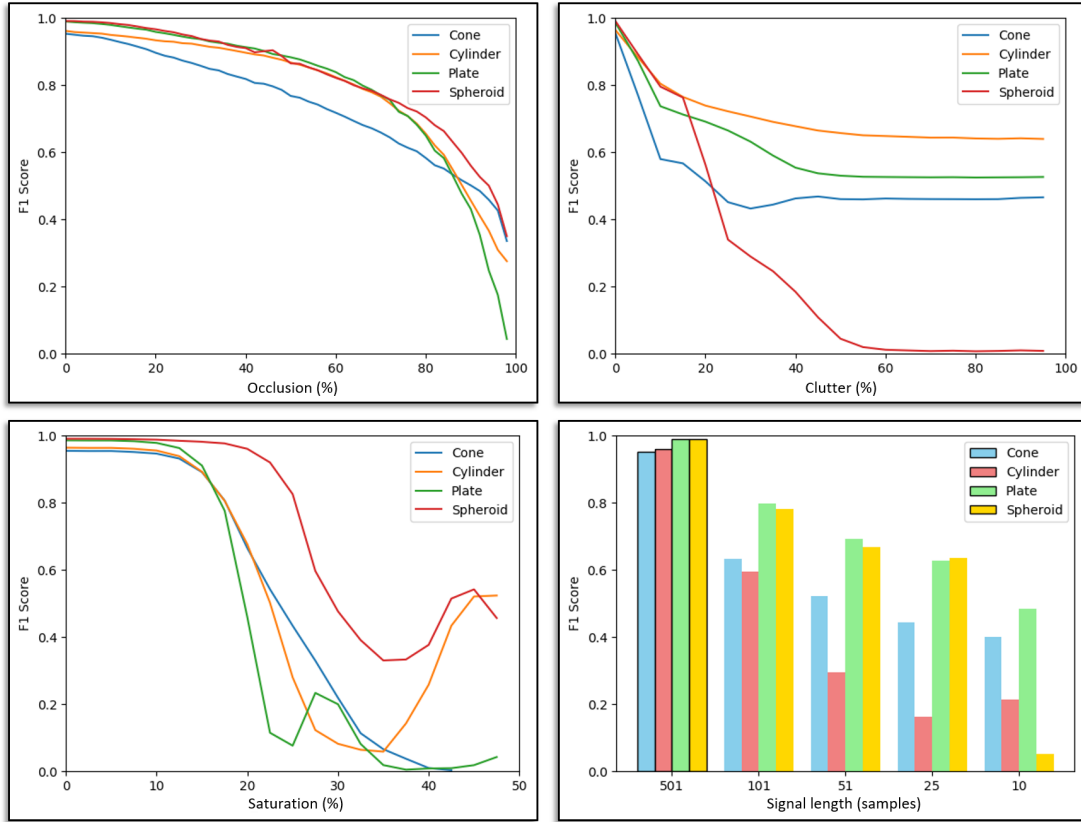


Figure 5.14: The single 18 layered residual network’s robustness performance is shown for several novel distortions. This benchmark is a way to compare a network robustness to realistic signal distortions found RCS systems. Signal occlusion, clutter, saturation, and subsampling are the realistic distortions used for this benchmark. Source: [5].

typically few variables are able to be controlled when creating real datasets. Simulated data is useful because very large datasets can be generated easily. Adjustments can be made one variable at a time and all parameters used to create that data is known at every timestep. The generative CNN called the refiner network described in Section 4 makes simulated data appear more like real data, shown in Figure 5.9. Using the refined data to train a small network on a subset of our A4 dataset results in a 3.5% accuracy improvement over training using the equivalent simulated data. For that test the only “realistic” feature added to the “real” data was noise. In Figure 5.9 we see that the refined signal seemingly adds noise to the simulated signal but maintains the structural elements of the signal.

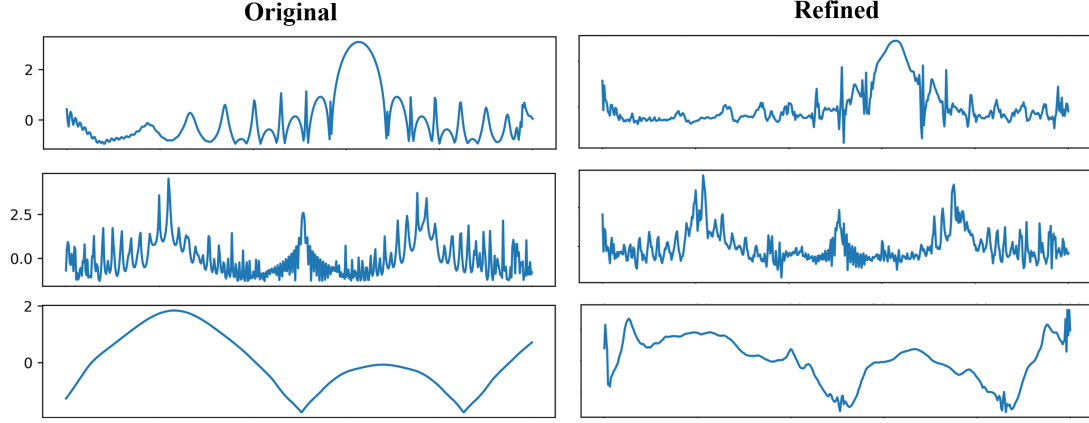


Figure 5.15: Some examples of signals pre and post refinement. The structure of the signal is maintained but pseudo noise is added to the original signal from the refiner network. Source: [5].

## 6 Discussion

To the best of our knowledge, we are the first to train convolutional neural networks to classify object shape from monostatic radar signals. We expand upon the MATLAB library POFacets to generate large datasets with a variety of selected parameters. Realistic motion, added noise, and Swerling dropout enhance the initial simulation generation. Utilizing the latest in deep learning architecture we create a 1D residual network capable of achieving test error results as low as 2-2.5% on our generated datasets. Our A4 dataset is augmented with an additional test and then evaluated with a siamese network architecture. The siamese CNN does perform as well in terms of accuracy but surpasses the performance of the single residual network in terms of average F1-score. The robustness of our CNN is then evaluated on signals with previously unseen realistic distortions. The single residual network performs well on signals with occlusion and subsampling but performs poorly on signals with clutter and saturation. We explored increasing the quality of the simulated signals using a state of the art refiner network. Deep learning models trained on the refined signals outperform models trained on the original simulated data.

## Chapter 6

### Conclusion

Camera-display communication and photographic steganography is a challenging problem with many interesting applications. In Chapter 2, we introduce the *Camera-Display Transfer Function (CDTF)* and propose two methods for online radiometric calibration. The presence of the CDTF is what distinguishes digital steganography from photographic steganography, a significantly more challenging problem. Chapter 3 proposes how a new class of color pairs called *differential metamers* can simultaneously reduce message recovery errors and visual obtrusiveness in photographic steganography. In Chapter 4, we construct a dataset of one million image pairs and use deep learning methods to model the CDTF for 25 camera-display pairs. We then learn a message embedding and recovery algorithm based on spacial gradients that requires no multi-frame synchronization between camera and display, a major practical barrier to real-world implementation. Finally, Chapter 5 extends computer vision techniques to the monostatic radar domain, where object shape is recognized from a noisy time-series signal.

What are the remaining problems associated with photographic steganography? Currently, the assumption is made that the boundaries of an imaged electronic display are known, but this problem has not been solved without tagging the physical display with fiducial markers or preprocessing the content images for feature extraction. The 2D barcode message structure used in this thesis is simple, but could be improved for robust message recovery under a variety of imaging situations. No formal study has been made quantifying the relationship between the robustness of message recovery and visual obtrusiveness. There are a number of compression algorithms used in various video codecs. Currently, there are no photographic steganography methods that are explicitly robust to each of these methods.



## Bibliography

- [1] S. Baluja, “Hiding images in plain sight: Deep steganography,” in *Advances in Neural Information Processing Systems*, 2017, pp. 2066–2076.
- [2] E. Wengrowski, W. Yuan, K. J. Dana, A. Ashok, M. Gruteser, and N. Mandayam, “Optimal radiometric calibration for camera-display communication,” in *Applications of Computer Vision (WACV), 2016 IEEE Winter Conference on*. IEEE, 2016, pp. 1–10.
- [3] E. Wengrowski, K. J. Dana, M. Gruteser, and N. Mandayam, “Reading between the pixels: Photographic steganography for camera display messaging,” in *Computational Photography (ICCP), 2017 IEEE International Conference on*. IEEE, 2017, pp. 1–11.
- [4] E. Wengrowski and K. Dana, “Light field messaging with deep photographic steganography,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2019.
- [5] E. Wengrowski, M. Purri, K. Dana, and A. Huston, “Deep convolutional neural networks as a method to classify rotating objects based on monostatic radar cross section,” *IET Radar, Sonar & Navigation*, 2019.
- [6] Y. Liu, J. Yang, and M. Liu, “Recognition of qr code with mobile phones,” in *2008 Chinese control and decision conference*. IEEE, 2008, pp. 203–206.
- [7] A. Ashok, M. Gruteser, N. Mandayam, J. Silva, M. Varga, and K. Dana, “Challenge: Mobile optical networks through visual mimo,” in *Proceedings of the sixteenth annual international conference on Mobile computing and networking*. ACM, 2010, pp. 105–112.
- [8] S. D. Perli, N. Ahmed, and D. Katabi, “Pixnet: Interference-free wireless links using lcd-camera pairs,” in *Proceedings of the sixteenth annual international conference on Mobile computing and networking*. ACM, 2010, pp. 137–148.
- [9] M. Varga, A. Ashok, M. Gruteser, N. Mandayam, W. Yuan, and K. Dana, “Visual mimo based led-camera communication applied to automobile safety,” in *Proceedings of the 9th international conference on Mobile systems, applications, and services*. ACM, 2011, pp. 383–384.
- [10] W. Yuan, K. Dana, M. Varga, A. Ashok, M. Gruteser, and N. Mandayam, “Computer vision methods for visual mimo optical system,” in *Computer Vision and Pattern Recognition Workshops (CVPRW), 2011 IEEE Computer Society Conference on*. IEEE, 2011, pp. 37–43.

- [11] A. Ashok, M. Gruteser, N. Mandayam, T. Kwon, W. Yuan, M. Varga, and K. Dana, "Rate adaptation in visual mimo," in *2011 8th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks*. IEEE, 2011, pp. 583–591.
- [12] W. Yuan, K. Dana, A. Ashok, M. Varga, M. Gruteser, and N. Mandayam, "Photographic steganography for visual mimo: A computer vision approach," in *IEEE Workshop on the Applications of Computer Vision (WACV)*, 2012, pp. 345–352.
- [13] W. Yuan, K. J. Dana, A. Ashok, M. Gruteser, and N. Mandayam, "Spatially varying radiometric calibration for camera-display messaging," in *2013 IEEE Global Conference on Signal and Information Processing*. IEEE, 2013, pp. 763–766.
- [14] A. Ashok, S. Jain, M. Gruteser, N. Mandayam, W. Yuan, and K. Dana, "Capacity of screen-camera communications under perspective distortions," *Pervasive and Mobile Computing*, vol. 16, pp. 239–250, 2015.
- [15] K. Jo, M. Gupta, and S. K. Nayar, "Disco: Display-camera communication using rolling shutter sensors," *ACM Transactions on Graphics (TOG)*, vol. 35, no. 5, p. 150, 2016.
- [16] T. Mitsunaga and S. K. Nayar, "Radiometric self calibration," in *Proceedings. 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No PR00149)*, vol. 1. IEEE, 1999, pp. 374–380.
- [17] P. Debevec and J. Malik, "Recovering high dynamic range radiance maps from photographs," *ACM SIGGRAPH*, pp. pp. 369–378, 1997.
- [18] S. K. Nayar and T. Mitsunaga, "High dynamic range imaging: spatially varying pixel exposures," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. pp. 472–479, 2000.
- [19] A. Cheddad, J. Condell, K. Curran, and P. Mc Kevitt, "Digital image steganography: Survey and analysis of current methods," *Signal processing*, vol. 90, no. 3, pp. 727–752, 2010.
- [20] P. Wayner, *Disappearing cryptography: information hiding: steganography and watermarking*. Morgan Kaufmann, 2009.
- [21] V. M. Potdar, S. Han, and E. Chang, "A survey of digital image watermarking techniques," in *INDIN'05. 2005 3rd IEEE International Conference on Industrial Informatics, 2005*. IEEE, 2005, pp. 709–716.
- [22] I. J. Cox, M. L. Miller, J. A. Bloom, and C. Honsinger, *Digital watermarking*. Springer, 2002, vol. 53.
- [23] N. F. Johnson, Z. Duric, and S. Jajodia, *Information Hiding: Steganography and Watermarking-Attacks and Countermeasures: Steganography and Watermarking: Attacks and Countermeasures*. Springer Science & Business Media, 2001, vol. 1.
- [24] F. A. Petitcolas, R. J. Anderson, and M. G. Kuhn, "Information hiding-a survey," *Proceedings of the IEEE*, vol. 87, no. 7, pp. 1062–1078, 1999.

- [25] P. Dong, J. G. Brankov, N. P. Galatsanos, Y. Yang, and F. Divoine, "Digital watermarking robust to geometric distortions," *IEEE Transactions on Image Processing*, vol. 14, no. 12, pp. 2140–2150, 2005.
- [26] A. Sangeetha, B. Gomathy, and K. Anusudha, "A watermarking approach to combat geometric attacks," in *2009 International Conference on Digital Image Processing*. IEEE, 2009, pp. 381–385.
- [27] J.-L. Dugelay, S. Roche, C. Rey, and G. Doërr, "Still-image watermarking robust to local geometric distortions," *IEEE transactions on image processing*, vol. 15, no. 9, pp. 2831–2842, 2006.
- [28] X.-y. Wang, L.-m. Hou, and J. Wu, "A feature-based robust digital image watermarking against geometric attacks," *Image and Vision Computing*, vol. 26, no. 7, pp. 980–989, 2008.
- [29] C.-Y. Lin, M. Wu, J. A. Bloom, I. J. Cox, M. L. Miller, and Y. M. Lui, "Rotation, scale, and translation resilient watermarking for images," *IEEE Transactions on image processing*, vol. 10, no. 5, pp. 767–782, 2001.
- [30] J. S. .Seo and C. D. Yoo, "Image watermarking based on invariant regions of scale-space representation," *Signal Processing, IEEE Transactions on*, vol. 54, no. 4, pp. 1537 – 1549, 2006.
- [31] F. Zou, H. Ling, X. Li, Z. Xu, and P. Li, "Robust image copy detection using local invariant feature," in *Multimedia Information Networking and Security, 2009. MINES '09. International Conference on*, vol. 1, 2009, pp. 57–61.
- [32] L. Yang and Z. Guo, "A robust video watermarking scheme resilient to spatial desynchronization and photometric distortion," in *Signal Processing, 2006 8th International Conference on*, vol. 4, 16-20 2006.
- [33] S. Mann and R. Picard, "On being undigital with digital cameras: Extending dynamic range by combining differently exposed pictures," *Proc. IST 46th annual conference*, pp. 422 – 428, 1995.
- [34] S. J. Kim and M. Pollefeys, "Robust radiometric calibration and vignetting correction," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 4, pp. 562 –576, april 2008.
- [35] A. Chakrabarti, D. Scharstein, and T. Zickler, "An empirical camera model for internet color vision," *British Machine Vision Conference*, 2009.
- [36] J.-Y. Lee, Y. Matsushita, B. Shi, I. S. Kweon, and K. Ikeuchi, "Radiometric calibration by rank minimization," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 35, no. 1, pp. 144 –156, jan. 2013.
- [37] M. D. Grossberg and S. K. Nayar, "What can be known about the radiometric response from images?" in *Proceedings of the 7th European Conference on Computer Vision-Part IV*, ser. ECCV '02. London, UK, UK: Springer-Verlag, 2002, pp. 189–205. [Online]. Available: <http://dl.acm.org/citation.cfm?id=645318.649359>

- [38] S. J. Kim, H. T. Lin, Z. Lu, S. Susstrunk, S. Lin, and M. S. Brown, “A new in-camera imaging model for color computer vision and its application,” in *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2012.
- [39] H. T. Lin, S. J. Kim, S. Susstrunk, and M. S. Brown, “Revisiting radiometric calibration for color computer vision,” *ICCV*, 2011.
- [40] Y. Xiong, K. Saenko, T. Darrell, and T. Zickler, “From pixels to physics: Probabilistic color de-rendering,” *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 358–365, 2012.
- [41] A. Mohan, G. Woo, S. Hiura, Q. Smithwick, and R. Raskar, “Bokode: imperceptible visual tags for camera based interaction from a distance,” in *SIGGRAPH*. ACM, 2009.
- [42] K. Kamijo, N. Kamijo, and G. Zhang, “Invisible barcode with optimized error correction,” in *Image Processing, 2008. IICIP 2008. 15th IEEE International Conference on*, oct. 2008, pp. 2036–2039.
- [43] J. Vucic, C. Kottke, S. Nerreter, K. Langer, and J. Walewski, “513 mbit/s visible light communications link based on dmt-modulation of a white led,” *Journal of Lightwave Technology*, vol. 28, no. 24, pp. 3512–3518, 2010.
- [44] W. Yuan, R. E. Howard, K. J. Dana, R. Raskar, A. Ashok, M. Gruteser, and N. Mandayam, “Phase messaging method for time-of-flight cameras,” in *Computational Photography (ICCP), 2014 IEEE International Conference on*. IEEE, 2014, pp. 1–8.
- [45] T. Hao, R. Zhou, and G. Xing, “Cobra: color barcode streaming for smartphone systems,” in *Proceedings of the 10th international conference on Mobile systems, applications, and services*. ACM, 2012, pp. 85–98.
- [46] A. Wang, S. Ma, C. Hu, J. Huai, C. Peng, and G. Shen, “Enhancing reliability to boost the throughput over screen-camera links,” in *Proceedings of the 20th annual international conference on Mobile computing and networking*. ACM, 2014, pp. 41–52.
- [47] W. Hu, J. Mao, Z. Huang, Y. Xue, J. She, K. Bian, and G. Shen, “Strata: layered coding for scalable visual communication,” in *Proceedings of the 20th annual international conference on Mobile computing and networking*. ACM, 2014, pp. 79–90.
- [48] W. Hu, H. Gu, and Q. Pu, “Lightsync: Unsynchronized visual communication over screen-camera links,” in *Proceedings of the 19th annual international conference on Mobile computing & networking*. ACM, 2013, pp. 15–26.
- [49] J. B. Peter and E. H. Adelson, “The laplacian pyramid as a compact image code,” *IEEE Transactions on Communications*, vol. 31, pp. 532–540, 1983.
- [50] W. Yuan, K. Dana, A. Ashok, M. Varga, M. Gruteser, and N. Mandayam, “Photographic steganography for visual mimo: A computer vision approach,” *IEEE Workshop on the Applications of Computer Vision (WACV)*, pp. 345–352, 2012.

- [51] V. Nguyen, Y. Tang, A. Ashok, M. Gruteser, K. Dana, W. Hu, E. Wengrowski, and N. Mandayam, “High-rate flicker-free screen-camera communication with spatially adaptive embedding,” in *IEEE INFOCOM*, vol. 2, 2016.
- [52] D. L. MacAdam, “Specification of small chromaticity differences,” *JOSA*, vol. 33, no. 1, pp. 18–26, 1943.
- [53] W. Commons, “Ciexy1931 macadam.png,” [https://en.wikipedia.org/wiki/File:CIExy1931\\_MacAdam.png](https://en.wikipedia.org/wiki/File:CIExy1931_MacAdam.png), 2015.
- [54] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.
- [55] W. Brown and D. MacAdam, “Visual sensitivities to combined chromaticity and luminance differences,” *JOSA*, vol. 39, no. 10, pp. 808–823, 1949.
- [56] G. Wyszecki and G. Fielder, “New color-matching ellipses,” *JOSA*, vol. 61, no. 9, pp. 1135–1152, 1971.
- [57] W. R. Brown, “Color discrimination of twelve observers,” *JOSA*, vol. 47, no. 2, pp. 137–143, 1957.
- [58] H. R. Davidson, “Calculation of color differences from visual sensitivity ellipsoids,” *JOSA*, vol. 41, no. 12, pp. 1052–1055, 1951.
- [59] A. B. Poirson, B. A. Wandell, D. C. Varner, and D. H. Brainard, “Surface characterizations of color thresholds,” *J. Opt. Soc. Am. A*, vol. 7, no. 4, pp. 783–789, Apr 1990.
- [60] G. Wyszecki, V. Stiles, and K. L. Kelly, “Color science: concepts and methods, quantitative data and formulas,” *Physics Today*, vol. 21, no. 6, pp. 83–84, 2009.
- [61] C. Noorlander and J. J. Koenderink, “Spatial and temporal discrimination ellipsoids in color space,” *J. Opt. Soc. Am.*, vol. 73, no. 11, pp. 1533–1543, Nov 1983.
- [62] G. D. Finlayson and P. Morovic, “Metamer sets,” *JOSA A*, vol. 22, no. 5, pp. 810–819, 2005.
- [63] N. Rudaz and R. D. Hersch, “Protecting identity documents by microstructure color differences,” *Journal of Electronic Imaging*, vol. 13, no. 2, pp. 315–323, 2004.
- [64] W. Yuan, K. Dana, A. Ashok, M. Gruteser, and N. Mandayam, “Dynamic and invisible messaging for visual mimo,” in *Applications of Computer Vision (WACV), 2012 IEEE Workshop on*. IEEE, 2012, pp. 345–352.
- [65] T. Li, C. An, A. Campbell, and X. Zhou, “Hilight: hiding bits in pixel translucency changes,” in *Proceedings of the 1st ACM MobiCom workshop on Visible light communication systems*. ACM, 2014, pp. 45–50.
- [66] A. Ashok, M. Gruteser, N. Mandayam, J. Silva, M. Varga, and K. Dana, “Challenge: Mobile optical networks through visual mimo,” in *Proceedings of the Sixteenth Annual International Conference on Mobile Computing and*

- Networking*, ser. MobiCom '10. New York, NY, USA: ACM, 2010, pp. 105–112. [Online]. Available: <http://doi.acm.org/10.1145/1859995.1860008>
- [67] L. Zhang, C. Bo, J. Hou, X.-Y. Li, Y. Wang, K. Liu, and Y. Liu, “Kaleido: You can watch it but cannot record it,” in *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking*. ACM, 2015, pp. 372–385.
  - [68] G. Woo, A. Lippman, and R. Raskar, “Vrcodes: Unobtrusive and active visual codes for interaction by exploiting rolling shutter,” in *Mixed and Augmented Reality (ISMAR), 2012 IEEE International Symposium on*. IEEE, 2012, pp. 59–64.
  - [69] A. Ashok, V. Nguyen, M. Gruteser, N. Mandayam, W. Yuan, and K. Dana, “Do not share!: invisible light beacons for signaling preferences to privacy-respecting cameras,” in *Proceedings of the 1st ACM MobiCom workshop on Visible light communication systems*. ACM, 2014, pp. 39–44.
  - [70] P. Luo, M. Zhang, Z. Ghassemlooy, H. Le Minh, H.-M. Tsai, X. Tang, L. Png, and D. Han, “Experimental demonstration of rgb led-based optical camera communications.”
  - [71] P. Hu, P. H. Pathak, X. Feng, H. Fu, and P. Mohapatra, “Colorbars: Increasing data rate of led-to-camera communication using color shift keying.”
  - [72] D. Kelly, “Sine waves and flicker fusion,” *Documenta Ophthalmologica*, vol. 18, no. 1, pp. 16–35, 1964.
  - [73] M. Grant and S. Boyd, “CVX: Matlab software for disciplined convex programming, version 2.1,” <http://cvxr.com/cvx>, Mar. 2014.
  - [74] M. C. Grant and S. P. Boyd, “Graph implementations for nonsmooth convex programs,” in *Recent advances in learning and control*. Springer, 2008, pp. 95–110, [http://stanford.edu/~boyd/graph\\_dcp.html](http://stanford.edu/~boyd/graph_dcp.html).
  - [75] <https://github.com/mathski/LFM>.
  - [76] K. J. Dana, “Capturing computational appearance: More than meets the eye,” *IEEE Signal Processing Magazine*, vol. 33, no. 5, pp. 70–80, 2016.
  - [77] R. Chandramouli and N. Memon, “Analysis of lsb based image steganography techniques,” in *Image Processing, 2001. Proceedings. 2001 International Conference on*, vol. 3. IEEE, 2001, pp. 1019–1022.
  - [78] A. Ashok, S. Jain, M. Gruteser, N. Mandayam, W. Yuan, and K. Dana, “Capacity of pervasive camera based communication under perspective distortions,” in *2014 IEEE International Conference on Pervasive Computing and Communications (PerCom)*, March 2014, pp. 112–120.
  - [79] P. Mirdehghan, W. Chen, and K. N. Kutulakos, “Optimal structured light à la carte,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 6248–6257.
  - [80] W. Yuan, K. J. Dana, M. Varga, A. Ashok, M. Gruteser, and N. B. Mandayam, “Computer vision methods for visual mimo optical system,” *CVPR 2011 WORKSHOPS*, pp. 37–43, 2011.

- [81] A. Wang, C. Peng, O. Zhang, G. Shen, and B. Zeng, “Inframe: Multiflexing full-frame visible communication channel for humans and devices,” in *Proceedings of the 13th ACM Workshop on Hot Topics in Networks*, ser. HotNets-XIII. New York, NY, USA: ACM, 2014, pp. 23:1–23:7. [Online]. Available: <http://doi.acm.org/10.1145/2670518.2673867>
- [82] A. Wang, Z. Li, C. Peng, G. Shen, G. Fang, and B. Zeng, “Inframe++: Achieve simultaneous screen-human viewing and hidden screen-camera communication,” in *Proceedings of the 13th Annual International Conference on Mobile Systems, Applications, and Services*, ser. MobiSys ’15, 2015, pp. 181–195.
- [83] T. Li, C. An, X. Xiao, A. T. Campbell, and X. Zhou, “Real-time screen-camera communication behind any scene,” in *Proceedings of the 13th Annual International Conference on Mobile Systems, Applications, and Services*, ser. MobiSys ’15. New York, NY, USA: ACM, 2015, pp. 197–211. [Online]. Available: <http://doi.acm.org/10.1145/2742647.2742667>
- [84] T. Morkel, J. H. Eloff, and M. S. Olivier, “An overview of image steganography,” in *ISSA*, 2005, pp. 1–11.
- [85] J. Fridrich and M. Goljan, “Practical steganalysis of digital images: state of the art,” in *Security and Watermarking of Multimedia Contents IV*, vol. 4675. International Society for Optics and Photonics, 2002, pp. 1–14.
- [86] H. Wang and S. Wang, “Cyber warfare: steganography vs. steganalysis,” *Communications of the ACM*, vol. 47, no. 10, pp. 76–82, 2004.
- [87] R. Chandramouli, M. Kharrazi, and N. Memon, “Image steganography and steganalysis: Concepts and practice,” in *International Workshop on Digital Watermarking*. Springer, 2003, pp. 35–49.
- [88] L. M. Marvel, C. G. Boncelet, and C. T. Retter, “Spread spectrum image steganography,” *IEEE Transactions on image processing*, vol. 8, no. 8, pp. 1075–1083, 1999.
- [89] N. F. Johnson and S. Jajodia, “Exploring steganography: Seeing the unseen,” *Computer*, vol. 31, no. 2, 1998.
- [90] O. Ronneberger, P. Fischer, and T. Brox, “U-net: Convolutional networks for biomedical image segmentation,” in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.
- [91] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, “Densely connected convolutional networks,” in *CVPR*, vol. 1, no. 2, 2017, p. 3.
- [92] G. Koch, R. Zemel, and R. Salakhutdinov, “Siamese neural networks for one-shot image recognition,” in *ICML Deep Learning Workshop*, vol. 2, 2015.
- [93] L. Pibre, P. Jérôme, D. Ienco, and M. Chaumont, “Deep learning for steganalysis is better than a rich model with an ensemble classifier, and is natively robust to the cover source-mismatch. arxiv preprint,” *arXiv preprint arXiv:1511.04855*, 2015.

- [94] L. Pibre, J. Pasquet, D. Ienco, and M. Chaumont, "Deep learning is a good steganalysis tool when embedding key is reused for different images, even if there is a cover sourcemismatch," *Electronic Imaging*, vol. 2016, no. 8, pp. 1–11, 2016.
- [95] Y. Qian, J. Dong, W. Wang, and T. Tan, "Deep learning for steganalysis via convolutional neural networks," in *Media Watermarking, Security, and Forensics 2015*, vol. 9409. International Society for Optics and Photonics, 2015, p. 94090J.
- [96] I. Khan, B. Verma, V. K. Chaudhari, and I. Khan, "Neural network based steganography algorithm for still images," in *Emerging Trends in Robotics and Communication Technologies (INTERACT), 2010 International Conference on*. IEEE, 2010, pp. 46–51.
- [97] S. Husien and H. Badi, "Artificial neural network for steganography," *Neural Computing and Applications*, vol. 26, no. 1, pp. 111–116, 2015.
- [98] J. Hayes and G. Danezis, "Generating steganographic images via adversarial training," in *Advances in Neural Information Processing Systems*, 2017, pp. 1951–1960.
- [99] X. Weng, Y. Li, L. Chi, and Y. Mu, "Convolutional video steganography with temporal residual modeling," *arXiv preprint arXiv:1806.02941*, 2018.
- [100] P. Wu, Y. Yang, and X. Li, "Stegnet: Mega image steganography capacity with deep convolutional network," *arXiv preprint arXiv:1806.06357*, 2018.
- [101] J. Zhu, R. Kaplan, J. Johnson, and L. Fei-Fei, "Hidden: Hiding data with deep networks," *arXiv preprint arXiv:1807.09937*, 2018.
- [102] R. Meng, S. G. Rice, J. Wang, and X. Sun, "A fusion steganographic algorithm based on faster r-cnn," *Computers, Materials & Continua*, vol. 55, no. 1, pp. 1–1, 2018.
- [103] S. Dong, R. Zhang, and J. Liu, "Invisible steganography via generative adversarial network," *arXiv preprint arXiv:1807.08571*, 2018.
- [104] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.
- [105] J. Johnson, A. Alahi, and L. Fei-Fei, "Perceptual losses for real-time style transfer and super-resolution," in *European Conference on Computer Vision*. Springer, 2016, pp. 694–711.
- [106] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. P. Aitken, A. Tejani, J. Totz, Z. Wang *et al.*, "Photo-realistic single image super-resolution using a generative adversarial network." in *CVPR*, vol. 2, no. 3, 2017, p. 4.
- [107] Y. Blau and T. Michaeli, "The perception-distortion tradeoff," in *Proc. 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, Utah, USA*, 2018, pp. 6228–6237.



- [108] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, “The unreasonable effectiveness of deep features as a perceptual metric,” *IEEE Conference on Computer Vision and Pattern Recognition*, 2018.
- [109] H. Talebi and P. Milanfar, “Learned perceptual image enhancement,” in *Computational Photography (ICCP), 2018 IEEE International Conference on*. IEEE, 2018, pp. 1–13.
- [110] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *arXiv preprint arXiv:1409.1556*, 2014.
- [111] L. A. Gatys, A. S. Ecker, and M. Bethge, “Image style transfer using convolutional neural networks,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2414–2423.
- [112] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, “Microsoft coco: Common objects in context,” in *European conference on computer vision*. Springer, 2014, pp. 740–755.
- [113] D. Kinga and J. B. Adam, “A method for stochastic optimization,” in *International Conference on Learning Representations (ICLR)*, vol. 5, 2015.
- [114] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv preprint arXiv:1412.6980*, 2014.
- [115] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, “Automatic differentiation in pytorch,” 2017.
- [116] S. Kanya, “Watermark dct,” <https://www.mathworks.com/matlabcentral/fileexchange/46866-watermark-dct>, 2014.
- [117] P. Swerling, “Probability of detection for fluctuating targets,” *IRE Transactions on Information Theory*, vol. 6, no. 2, pp. 269–308, 1960.
- [118] E. Jones, T. Oliphant, and P. Peterson, “{SciPy}: open source scientific tools for {Python},” 2014.
- [119] F. Chatzigeorgiadis, “Development of code for a physical optics radar cross section prediction and analysis application,” Ph.D. dissertation, Monterey California. Naval Postgraduate School, 2004.
- [120] P. Touzopoulos, D. Boviatsis, and K. C. Zikidis, “3d modelling of potential targets for the purpose of radar cross section (rcs) prediction: Based on 2d images and open source data,” in *Military Technologies (ICMT), 2017 International Conference on*. IEEE, 2017, pp. 636–642.
- [121] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Advances in neural information processing systems*, 2012, pp. 1097–1105.
- [122] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial nets,” in *Advances in neural information processing systems*, 2014, pp. 2672–2680.

- [123] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1–9.
- [124] G. Huang, Z. Liu, K. Q. Weinberger, and L. van der Maaten, "Densely connected convolutional networks," *arXiv preprint arXiv:1608.06993*, 2016.
- [125] G. Kouemou, "Radar target classification technologies," in *Radar Technology*. InTech, 2010.
- [126] K. J. Lang, A. H. Waibel, and G. E. Hinton, "A time-delay neural network architecture for isolated word recognition," *Neural networks*, vol. 3, no. 1, pp. 23–43, 1990.
- [127] J. A. Anderson, M. T. Gately, P. A. Penz, and D. R. Collins, "Radar signal categorization using a neural network," *Proceedings of the IEEE*, vol. 78, no. 10, pp. 1646–1657, 1990.
- [128] I. Jordanov and N. Petrov, "Sets with incomplete and missing data in radar signal classification," in *Neural Networks (IJCNN), 2014 International Joint Conference on*. IEEE, 2014, pp. 218–224.
- [129] I. Jordanov, N. Petrov, and A. Petrozziello, "Supervised radar signal classification," in *Neural Networks (IJCNN), 2016 International Joint Conference on*. IEEE, 2016, pp. 1464–1471.
- [130] A. Soto, A. Mendoza, and B. C. Flores, "Optimization of neural network architecture for classification of radar jamming fm signals," in *Radar Sensor Technology XXI*, vol. 10188. International Society for Optics and Photonics, 2017, p. 101881H.
- [131] A. Mendoza, A. Soto, and B. C. Flores, "Classification of radar jammer fm signals using a neural network," in *Radar Sensor Technology XXI*, vol. 10188. International Society for Optics and Photonics, 2017, p. 101881G.
- [132] Y. Hara, R. G. Atkins, S. H. Yueh, R. T. Shin, and J. A. Kong, "Application of neural networks to radar image classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 32, no. 1, pp. 100–109, 1994.
- [133] Y. Zhang and L. Wu, "Crop classification by forward neural network with adaptive chaotic particle swarm optimization," *Sensors*, vol. 11, no. 5, pp. 4721–4743, 2011.
- [134] S. Park, J. P. Hwang, E. Kim, H. Lee, and H. G. Jung, "A neural network approach to target classification for active safety system using microwave radar," *Expert Systems with Applications*, vol. 37, no. 3, pp. 2340–2346, 2010.
- [135] A. Rahman, E. Yavari, V. M. Lubecke, and O.-B. Lubecke, "Noncontact doppler radar unique identification system using neural network classifier on life signs," in *Biomedical Wireless Technologies, Networks, and Sensing Systems (BioWireless), 2016 IEEE Topical Conference on*. IEEE, 2016, pp. 46–48.

- [136] P. Szymczyk and M. Szymczyk, "Classification of geological structure using ground penetrating radar and laplace transform artificial neural networks," *Neurocomputing*, vol. 148, pp. 354–362, 2015.
- [137] C. Kabakchiev, V. Behar, I. Garvanov, D. Kabakchieva, and H. Rohling, "Detection, parametric imaging and classification of very small marine targets emerged in heavy sea clutter utilizing gps-based forward scattering radar," in *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*. IEEE, 2014, pp. 793–797.
- [138] J. Lundén and V. Koivunen, "Deep learning for hrrp-based target recognition in multistatic radar systems," in *Radar Conference (RadarConf), 2016 IEEE*. IEEE, 2016, pp. 1–6.
- [139] D. A. Morgan, "Deep convolutional neural networks for atr from sar imagery," *Proceedings of the Algorithms for Synthetic Aperture Radar Imagery XXII, Baltimore, MD, USA*, vol. 23, p. 94750F, 2015.
- [140] Y. Kim and T. Moon, "Human detection and activity classification based on micro-doppler signatures using deep convolutional neural networks," *IEEE Geoscience and Remote Sensing Letters*, vol. 13, no. 1, pp. 8–12, 2016.
- [141] M. Gong, J. Zhao, J. Liu, Q. Miao, and L. Jiao, "Change detection in synthetic aperture radar images based on deep neural networks," *IEEE transactions on neural networks and learning systems*, vol. 27, no. 1, pp. 125–138, 2016.
- [142] E. Mason, B. Yonel, and B. Yazici, "Deep learning for radar," in *Radar Conference (RadarConf), 2017 IEEE*. IEEE, 2017, pp. 1703–1708.
- [143] X. Zhang and Y. LeCun, "Text understanding from scratch," *arXiv preprint arXiv:1502.01710*, 2015.
- [144] S. Kiranyaz, T. Ince, and M. Gabbouj, "Real-time patient-specific ecg classification by 1-d convolutional neural networks," *IEEE Transactions on Biomedical Engineering*, vol. 63, no. 3, pp. 664–675, 2016.
- [145] P. Stinco, M. S. Greco, F. Gini, and M. La Manna, "Non-cooperative target recognition in multistatic radar systems," *IET Radar, Sonar & Navigation*, vol. 8, no. 4, pp. 396–405, 2013.
- [146] Z. Mathews, L. Quiriconi, U. Böniger, C. Schüpbach, and P. Weber, "Learning multi-static contextual target signatures," in *Radar Conference (RadarConf), 2017 IEEE*. IEEE, 2017, pp. 1568–1572.
- [147] A. Shrivastava, T. Pfister, O. Tuzel, J. Susskind, W. Wang, and R. Webb, "Learning from simulated and unsupervised images through adversarial training," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 3, no. 4, 2017, p. 6.
- [148] Z. Xin, W. Ying, and Y. Bin, "Signal classification method based on support vector machine and high-order cumulants," *Wireless Sensor Network*, vol. 2, no. 01, p. 48, 2010.

- [149] M. F. Byl, J. T. Demers, and E. A. Rietman, “Using a kernel adatron for object classification with rcs data,” *arXiv preprint arXiv:1005.5337*, 2010.
- [150] L. Zhang, W. Zhou, and L. Jiao, “Wavelet support vector machine,” *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 34, no. 1, pp. 34–39, 2004.
- [151] J. Bromley, I. Guyon, Y. LeCun, E. Säckinger, and R. Shah, “Signature verification using a” siamese” time delay neural network,” in *Advances in Neural Information Processing Systems*, 1994, pp. 737–744.
- [152] Y. Sun, X. Wang, and X. Tang, “Deep learning face representation from predicting 10,000 classes,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 1891–1898.
- [153] I. J. Goodfellow, J. Shlens, and C. Szegedy, “Explaining and harnessing adversarial examples,” *arXiv preprint arXiv:1412.6572*, 2014.