information

Article

Flying Projectile Attitude Determination from Ground-Based Monocular Imagery with a Priori Knowledge

Huamei Chen [1], Zhigang Zhu [2], Hao Tang [3], Erik Blasch [4], Khanh D. Pham [5] and Genshe Chen [1,*]

[1] Intelligent Fusion Technology, Inc., Germantown, MD 20874, USA; huamei.chen@intfusiontech.com
[2] Department of Computer Science, City University of New York, New York, NY 10031, USA; zzhu@ccny.cuny.edu
[3] Department of Computer Information Systems, Borough of Manhattan Community College, City University of New York, New York, NY 10007, USA; htang@bmcc.cuny.edu
[4] Air Force Research Laboratory, Arlington, VA 22203, USA; erik.blasch.1@us.af.mil
[5] Air Force Research Laboratory, Kirtland Air Force Base, Albuquerque, NM 87117, USA; khanh.pham.1@us.af.mil
* Correspondence: gchen@intfusiontech.com; Tel.: +1-240-481-5397

Abstract: This paper discusses using ground-based imagery to determine the attitude of a flying projectile assuming prior knowledge of its external geometry. It presents a segmentation-based approach to follow the object and evaluates it quantitatively with simulated data and qualitatively with both simulated and real data. Two experimental cases are considered: One assumes reliable target distance measurement from an auxiliary range sensor, while the other assumes no range information. The results show that in the case of an unknown projectile–camera distance, with projectile dimensions of 1.378 m and 0.08 m in length and diameter, the estimated distance, in-plane location, and pitch angle accuracies are about 50 m, 0.15 m, and 6 degrees, respectively. Yaw angle estimation is ambiguous. In the second case, assuming that the projectile–camera distance is known resolves the ambiguity of yaw estimation, resulting in accuracies of about 0.15 m, 3 degrees, and 20 degrees for in-plane location, pitch, and yaw angles, respectively. These accuracies were normalized to a 1-km projectile–camera distance.

Keywords: image segmentation; projectile attitude determination; 2D TETRIS

check for updates

Citation: Chen, H.; Zhu, Z.; Tang, H.; Blasch, E.; Pham, K.D.; Chen, G. Flying Projectile Attitude Determination from Ground-Based Monocular Imagery with a Priori Knowledge. *Information* **2024**, *15*, 201. https://doi.org/10.3390/info15040201

Academic Editors: Francesco Camastra and Kaihua Zhang

Received: 26 February 2024
Revised: 30 March 2024
Accepted: 31 March 2024
Published: 4 April 2024

1. Introduction

The photogrammetric determination of target attitude/object pose is a well-studied machine vision problem. The target attitude can be efficiently solved when given sufficiently resolved imagery and benign backgrounds. However, for ground-based imagery of objects in flight, such as aircraft, birds, or projectiles, the resolution can be marginal, and environments are rarely benign, i.e., there are varying lighting conditions, low signal-to-noise ratios (SNRs), cluttered backgrounds, and poor target contrast. Fortunately, for cooperative tests, the external geometry of airborne objects can be known a priori to whatever accuracy is desired. Using geometry encourages the development of a model-based method for estimating the attitude of a maneuvering airborne object. However, due to the intrinsic limitation of the considered application, detailed target information, such as color, texture, and depth, as exploited in many robot applications [1,2], are either not available or cannot be reliably exploited to match the projection of the 3D target model with the measured scene image. For example, what can be observed in ground-based imagery of projectiles in flight is the rough shape of the target. Using the shape causes one to resort to a segmentation technique, along with the development of effective similarity measures that are independent of color, gradient, and depth information, thereby discovering the matched target mask rendered from the 3D target model to determine the projectile attitude.

This study considers a scenario where a robust tracking module is in place to capture images of a flying projectile. It describes a segmentation-based flying projectile attitude determination workflow using ground-based monocular imagery with a priori knowledge in terms of an exact 3D projectile model. This is a practical scenario in the fields of testing and evaluation. The developed workflow is given in Figure 1, and it consists of three modules: detection and segmentation (DS), profile matching (PM), and shape-guided segmentation (SS). The DS module aims at detecting and segmenting the flying projectile in the current frame and produces an initial segmentation mask. Next, the PM module estimates the target attitude based on the segmentation mask, followed by searching for the best match between the segmentation mask and the rendered projectile profile. The result is an estimate of the projectile attitude. Since the initial mask is obtained through blind segmentation, it is expected that there is room for segmentation improvement, e.g., by incorporating the shape information into a segmentation algorithm. To this end, the rendered profile is exploited as a shape prior in the SS module to update the segmentation mask, and it is sent back to the PM module for refined projectile attitude estimation. Specifically, in this study, we compare three shape-guided image segmentation methods in the SS module: a modified Gradient Vector Flow Active Shape Model (GVF-ASM), a novel Shape-Incorporated Segmentation with the GraphCut criterion (SIS-Cut), and 2D Template Transformer Networks for Image Segmentation (TETRIS). GVF-ASM is devised by combining the active shape model (ASM) [3] into the gradient vector field Snake [4]. SIS-Cut is inspired by the criterion adopted in graph cut segmentation [5], and 2D TETRIS is a 2D implementation of TETRIS [6]. TETRIS is a machine-learning-based approach for the segmentation of coronary lumen structures in 3D cardiac computed tomography. Since it is very challenging, if not impossible, to obtain the ground-truth attitude of a flying projectile, simulated flying projectile imagery is needed to quantitatively evaluate the Imagery-Based Projectile Attitude Detection and Estimation (IMADE) solution and to generate the required training data, and it turns out that very few annotated frames are required to train the models employed in the DS and SS modules.

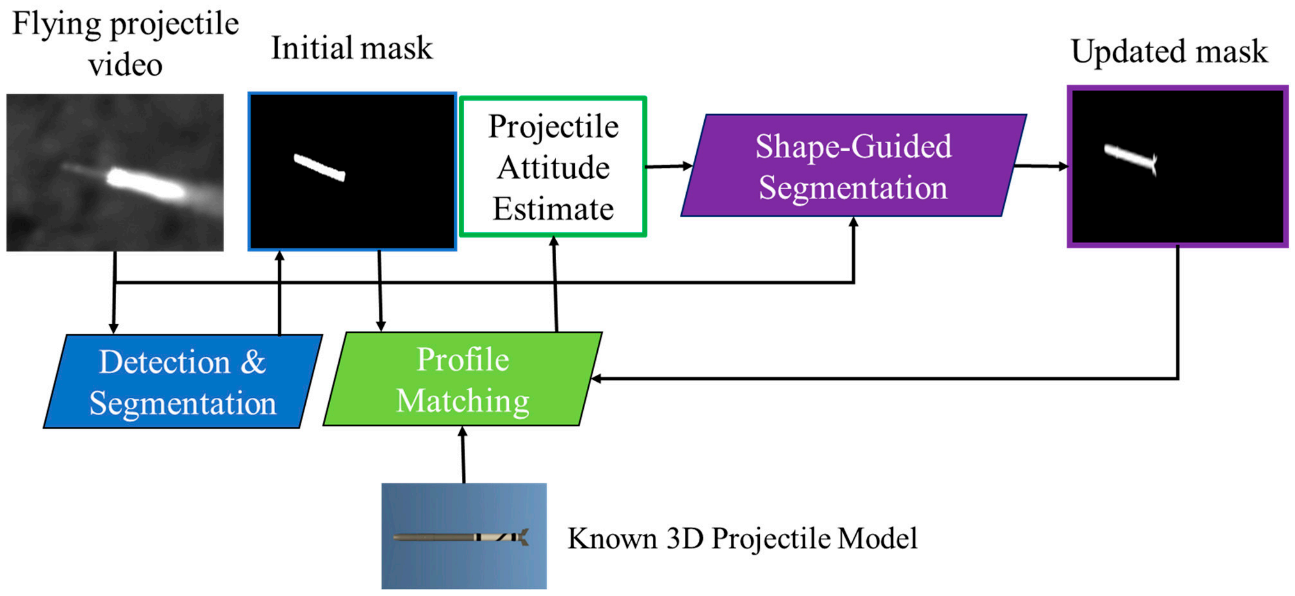


Figure 1. The developed workflow of segmentation-based flying projectile attitude determination using ground-based imagery with a priori knowledge.

IMADE uses a 3D flying projectile simulation program to generate synthetic ground-based flying projectile imagery using the Unity 3D game engine, version 2019.2.0. The complete 3D flying projectile simulation program includes the following features: (1) a Unity software application that renders realistic projectile images with different configurations. The application provides a user-friendly graphical user interface (GUI) that allows users to easily configure projectile models and complex environments with different physics characteristics. (2) The program also includes various textured 3D projectile models that are available for users to select for 3D projectile simulations and benchmark data generation, (3) a ground-truth data generation function that records real-time ground-truth annotation data, including six-DoF motion data of the projectile captured with multiple virtual cameras simultaneously, 2D semantic segmentation masks, and 3D bounding box annotations of the projectile, allowing users and collaborators to test and evaluate against state-of-the-art image segmentation approaches, and (4) a data parsing program that parses the metadata of the dataset generated by the above programs.

This paper is organized as follows. Section 2 provides a brief review of the literature on model-based projectile attitude determination from monocular images. The developed 3D flying projectile simulation program is described in Section 3. The implementations of the modules shown in Figure 1 are provided in Section 4. Section 5 describes the datasets used in the experiments and the experimental results. Section 6 lists conclusions with future directions.

2. Literature Review on Model-Based Target Attitude Determination

Attitude determination, which is commonly known as pose estimation, is an important step in applications such as aerial refueling [7], unmanned aerial vehicle (UAV) docking [8], and many industrial intelligence applications [9].

A closely related IMADE application that has been extensively studied is spacecraft attitude determination. Spacecraft attitude determination is critical for many essential tasks in space missions, e.g., formation flying, on-orbit servicing, and debris removal [10]. In general, the quality of the imagery captured for spacecraft attitude determination is considerably higher than that considered in this work due to the close-range nature of the tasks considered. A comparison of the typical actual and simulated images considered in spacecraft attitude determination and the flying projectile attitude determination considered in this study is provided in Figure 2, where (a) shows three actual camera images from the Spacecraft Pose Estimation Dataset (SPEED) [11], (b) depicts three synthetic images from the SPEED training set, (c) presents three real images of a flying projectile considered in the IMADE work, and (d) illustrates three simulated projectile images generated with the IMADE simulation program. To the best of our knowledge, no prior work has considered the problem of flying projectile attitude determination from ground-based imagery. Therefore, this section provides a brief review of the literature on the general approaches to estimating the 6D poses of objects.

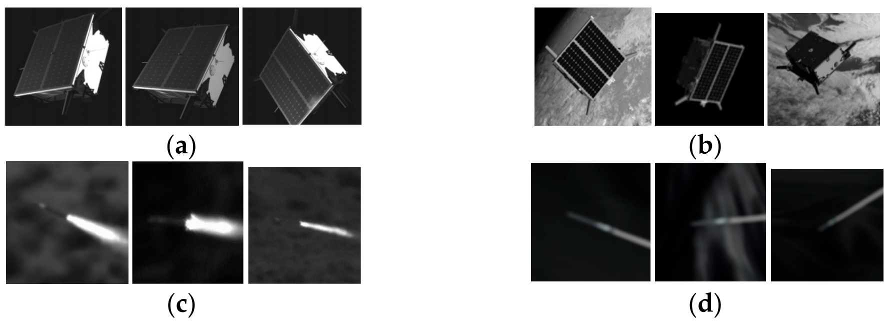


Figure 2. A comparison of typical images for spacecraft pose estimation and the flying projectile images considered in this work. (**a**,**c**) Actual images; (**b**,**d**) synthetic images. See the text for more information.

Detecting objects and estimating their poses in 3D have been extensively studied in robot manipulation [12] and augmented reality [13,14]. Various existing methods can be categorized from different perspectives. One such perspective is based on the data source, including options such as RGB-D cameras, dual RGB cameras [7,15], single RGB cameras, and others. These methods utilize RGB images and registered depth data (RGB-D) [1,2,16], only depth data [7,15,17], or only RGB images. Most recent works focus on using only RGB images due to the limitations imposed by depth sensors, i.e., the depth readings are not reliable in an outdoor environment. In what follows, methods that involve only RGB images are reviewed.

2.1. Template-Based Methods

Early object pose estimation research is cast as a template-based 3D object detection problem [18], and each template corresponds to a specific object pose. Thus, a large number of templates are required, and only coarse pose estimation is achieved [18–21]. For example, to detect an object under full coverage of viewpoints, about 2000 templates are generated [20]. The approaches that convert the 6D pose estimation problem into a pose classification problem by discretizing the pose space can be considered *template-based approaches* if one considers each individual class as a template. The work presented in [22] generates templates synthetically rather than using real annotated data, which requires expert knowledge and a complex setup [23]. In this way, much finer viewpoints (~20,000) are sampled to create templates for each object of interest. To generalize the synthetic templates to work with real object images, the authors of [23] proposed an augmented autoencoder (AAE) with domain randomization (DR) to reconstruct synthetically generated objects from real detected objects.

2.2. Keypoint-Based Methods

Keypoint-based methods, which are currently the most accurate and dominant approaches for pose estimation, adopt a two-stage pipeline: They first detect a few 2D keypoints of an object in an RGB image and match them with the keypoints from the 3D object model. Once a *keypoint-based* correspondence is established, an efficient Perspective-n-Point problem (PnP) algorithm [24] can be applied to find the transformation between the object coordinate system and the camera coordinate system. Traditional keypoint-based methods directly utilize point [25] or corner [19] features, edge features [26], or a combination of both [27] from 2D images to derive the points to match the corresponding 3D points in the object model. The PnP is a relatively simple approach for sufficiently textured objects where handcrafted features show promise in identifying the keypoints [28,29].

For textureless objects, recent works resort to machine-learning-based approaches to predicting the keypoints and establishing correspondences. The BB8 approach (bounding box using eight corners) [30] uses segmentation to identify image regions that contain objects and regresses the keypoints, which are defined as the 2D projections of the corners of an object's 3D bounding box from the segmented image regions. Yolo6D [31] adopted a similar approach but without resorting to segmentation. Another category of methods [32,33] uses heatmaps predicted from image patches to represent keypoints to address the issue of partial occlusion. Each heatmap is a 2D Gaussian distribution centered at the keypoint location with a small standard deviation. To additionally handle the case of truncated objects, a pixel-wise voting network (PVNet) [34] was proposed to learn a vector-field representation for robust 2D keypoint localization.

2.3. Direct Pose Estimation

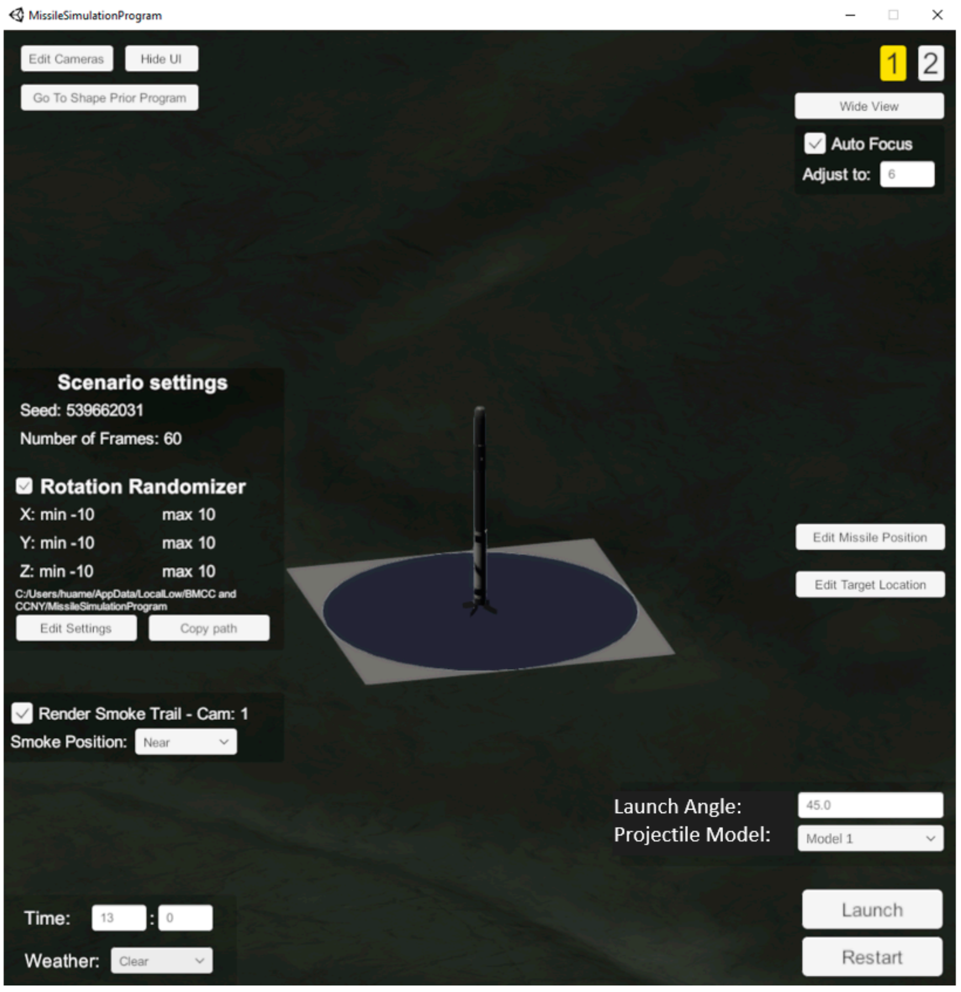
Unlike keypoint-based methods, which involve a two-stage pipeline, direct pose estimation approaches aim at directly regressing the 6D pose [35–37].

LieNet [35] detects and segments object instances in an image analogous to modern instance segmentation networks, such as a mask region-based convolutional neural network (R-CNN), but it contains a novel additional sub-network for 6D pose estimation. LieNet estimates the rotation matrix of an object by regressing a Lie-algebra-based rotation representation and estimates the translation vector by predicting the distance of the object from the camera center. Deep Iterative Matching (DeepIM) [36] is a 6D pose refinement approach. Given an initial pose estimation, DeepIM is trained to predict a relative pose transformation to iteratively refine the pose.

SSD-6D [37] extends the Single-Shot MultiBox Detector (SSD) by additionally outputting viewpoint classification and in-plane classification, which are used to directly estimate the 6D object pose. In [38], an end-to-end 6D pose estimation CNN with three heads was developed to predict the object class, estimate the object center, and regress the quaternion representation of object orientation.

3. Synthetic Data Generation

The IMADE simulation program was developed to facilitate the development of the a proposed solution, including through the generation of ground-based flying projectile imagery and ground-truth annotations. Figure 3 shows the main interface of the IMADE simulation program developed using the Unity 3D game engine. It includes background models, 3D projectile models, and camera models, which are briefly described in the following.



✓ Auto Focus
Adjust to: 6

Scenario settings
Seed: 539662031
Number of Frames: 60

✓ Rotation Randomizer

X: min -10	max 10
Y: min -10	max 10
Z: min -10	max 10

C:/Users/huame/AppData/LocalLow/BMCC and CCNY/MissileSimulationProgram

✓ Render Smoke Trail - Cam: 1
Smoke Position: Near

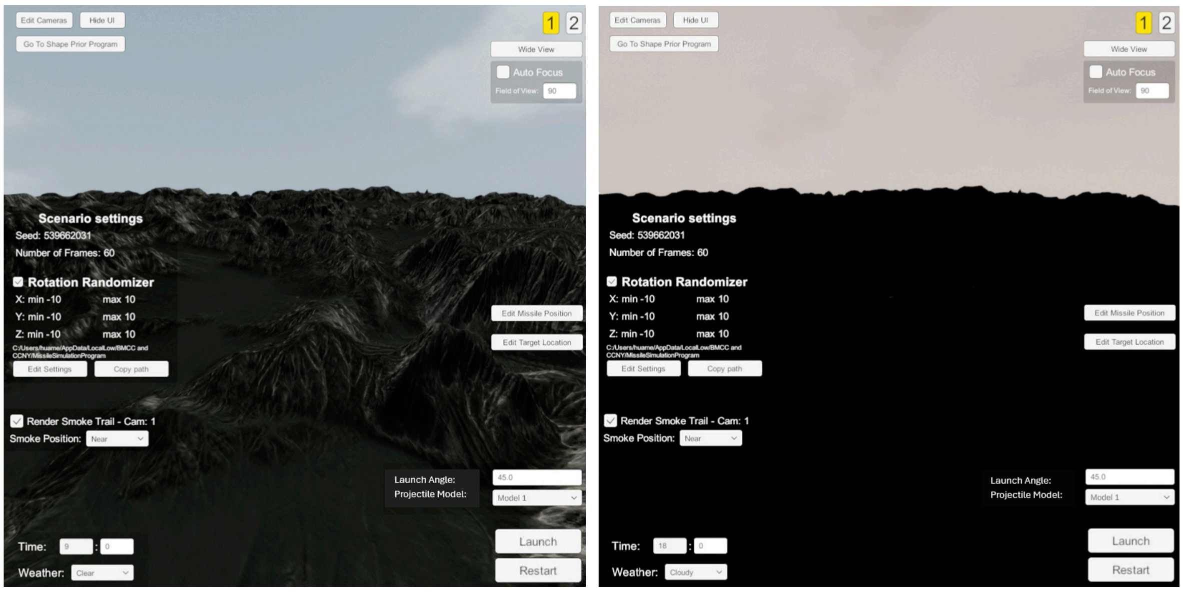
Launch Angle: 45.0
Projectile Model: Model 1

Time: 13 : 0
Weather: Clear

Figure 3. Flying projectile simulation program interface.

3.1. Background Models

The simulation program includes a 3D terrain model that allows users to configure different time and weather conditions, including cloudy and clear skies. Figure 4 shows the data that are captured at 9 a.m. with a clear sky and 6 p.m. with a cloudy sky.



Scenario settings
Seed: 539662031
Number of Frames: 60
Rotation Randomizer
X: min -10 max 10
Y: min -10 max 10
Z: min -10 max 10
C:/Users/huame/AppData/LocalLow/BMCC and CCNY/MissileSimulationProgram
Render Smoke Trail - Cam: 1
Smoke Position: Near
Field of View: 90
Launch Angle: 45.0
Projectile Model: Model 1
Time: 9 : 0
Weather: Clear

Scenario settings
Seed: 539662031
Number of Frames: 60
Rotation Randomizer
X: min -10 max 10
Y: min -10 max 10
Z: min -10 max 10
C:/Users/huame/AppData/LocalLow/BMCC and CCNY/MissileSimulationProgram
Render Smoke Trail - Cam: 1
Smoke Position: Near
Field of View: 90
Launch Angle: 45.0
Projectile Model: Model 1
Time: 18 : 0
Weather: Cloudy

(**a**) (**b**)

Figure 4. (**a**) The left image shows a scene captured at 9 a.m. with a clear sky, and (**b**) the right image shows a scene captured at 6 p.m. with a cloudy sky.

3.2. 3D Projectile Models

Seven projectile models have been created using the C4D modeling program and imported into the Unity 3D game engine. Users are allowed to choose any projectile model to generate benchmark data or shape prior images. The C4D program allows users to configure the position of a projectile's launch platform and destination, as well as its launch angle to simulate various projectile trajectories. The projectile's initial velocity and trajectory are calculated with a projectile model based on the initial and destination positions, along with the launch angle.

3.3. Camera Models

The IMADE simulation program allows users to configure multiple (up to five) virtual cameras to record projectile simulations at different locations using the Unity Perception library. The program provides two camera models, a wide field of view (FOV) without tracking the projectile, and a focused view with tracking the projectile. The projectile is rendered at the center of the images.

3.4. Ground-Truth Annotation

The IMADE simulation program generates ground-truth annotations using the Unity Perception program, which is a third-party asset for Unity. It is designed to create randomized scenarios, label objects, and capture images for the purpose of being fed to a machine learning model. There are many settings that allow users to adjust Perception to their needs. These settings include changing the capture speed, the duration for which Perception captures, what objects need labels, what objects are seen/unseen by Perception, etc. Furthermore, Perception creates JSON (JavaScript Object Notation) files for every image captured with much of the necessary information and ground-truth data, and it allows users to add other information. Since the Unity Perception program stores projectile and camera data in JSON files, we developed a Python program to parse ground-truth data and convert them into a CSV (comma-separated values) file. To parse the ground-truth data generated following the projectile simulation, we used the Unity Dataset Insights package.

4. Approaches

The details of the modules shown in Figure 1 are given in this section.

4.1. Detection and Segmentation Module

The first step in the workflow shown in Figure 1 is a module for target detection and segmentation. Unlike the double-thresholding-based method presented in [39], IMADE employs MaskR-CNN [40], a state-of-the-art instance segmentation approach developed for simultaneous object instance detection, classification, and segmentation. It was built by extending the Faster R-CNN architecture via a branch for predicting an object mask in parallel with the existing branch for bounding box recognition. Mask R-CNN takes the same two-stage procedure that Faster R-CNN takes. The first stage is a Region Proposal Network (RPN), which is identical to that included in Faster R-CNN, and the second stage, in addition to predicting the class and box offset, also outputs the binary mask for each region of interest.

4.2. Profile-Matching Module

The purpose of the profile-matching (PM) module is to determine the target attitude from a given mask detected and segmented with Mask R-CNN or the mask resulting from the SS module. PM additionally renders the projectile profile from the determined target attitude to facilitate refined segmentation in the SS module, as described in Figure 5. There are two components in the PM module: mask-based pose estimation and profile matching, which involves differentiable rendering and optimization with respect to the pose parameters. The inputs of the PM module are the segmentation mask (the mask detected with Mask R-CNN or the segmentation masks resulting from the SS module) and the 3D projectile model.

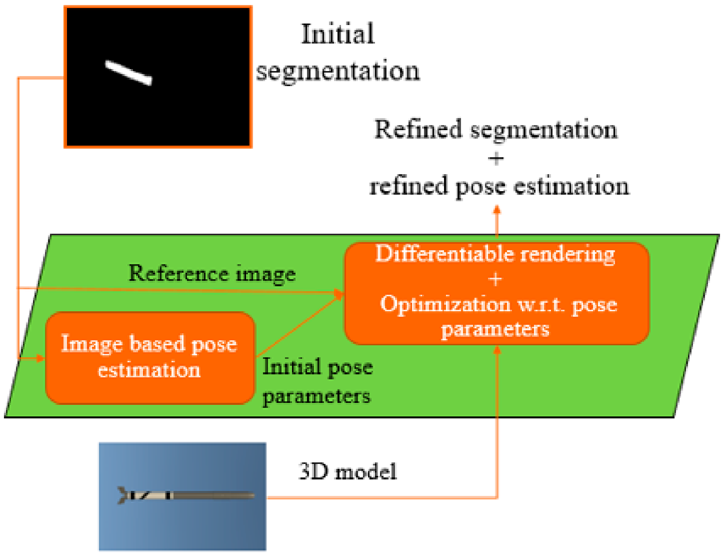
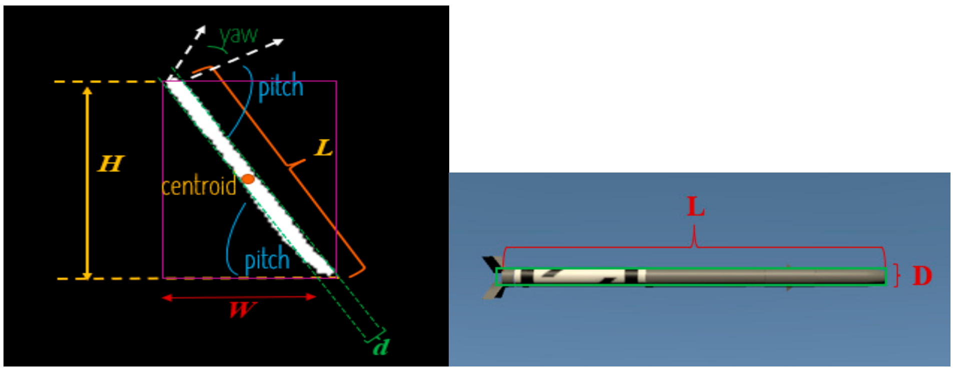

Initial segmentation
Refined segmentation + refined pose estimation
Reference image
Differentiable rendering + Optimization w.r.t. pose parameters
Image based pose estimation
Initial pose parameters
3D model

Figure 5. The profile-matching module.

4.2.1. Mask-Based Pose Estimation

Given the mask of a flying projectile, its pose can be estimated from the knowledge of the projectile dimensions and the intrinsic parameters of the viewing camera, as explained with the help of the plots given in Figure 6.



(**a**) (**b**)

Figure 6. Mask-based pose estimation.

Pitch and Yaw Estimation

Shown in Figure 6a is a mask detected with Mask R-CNN with the pitch and yaw indicated in the plot. With the quantities H, W, and L for the height, width, and length, respectively, as shown in Figure 6a, Equations (1) and (2) relate the geometric parameters to the pitch and yaw.

$$H = L \cdot \sin(pitch) \tag{1}$$

$$W = L \cdot cos(pitch) \cdot \sin(yaw) \tag{2}$$

where H and W are measured from the mask image, and they indicate the height and width of the bounding box of the mask. L is the length of the projectile as seen with a 0° pitch and ±90° yaw, as illustrated in Figure 6b. L can be precisely computed from the projectile dimensions, the intrinsic parameters of the camera, and the distance between the projectile and the viewing camera. Once H, W, and L are obtained, the *pitch* and *yaw* can be obtained by solving Equations (1) and (2). Alternatively, when the centroid of the projectile coincides with the image center, the pitch can be directly estimated as the angle between the projectile mask and the horizontal axis, as indicated in Figure 6a. Though this is the case considered in the simulation data, it may not be always true in real situations (see Figure 12 for real images of a flying projectile where the projectile centers are typically different from the image centers).

Ambiguity of Yaw Estimation

The ambiguity of yaw estimation is illustrated in Figure 7, which shows that two rendered projectiles with different yaw angles may correspond to the same detection mask. The ambiguity results from the fact that only a rough segmentation mask can be obtained from Mask R-CNN due to the marginal resolution of the ground-based imagery, as well as the possible occlusion from the exhaust plume. Simply put, given the detection mask, one cannot uniquely determine the value of the yaw. Specifically, if $yaw = -90 + y$, then from Equation (2),

$$-sin(yaw) = -sin(y - 90) = cos(y) = \frac{-W}{L \cdot cos(pitch)} \tag{3}$$

Since the cosine is an even function, one knows that $yaw = -90 - y$ is also a solution. Similarly, when $yaw = 90 + y$, its segmentation mask is indistinguishable from the one resulting from $yaw = 90 - y$.

To solve this issue of yaw estimation, the distance between the viewing camera and the target projectile, which is denoted as Tz, is crucial. More accurately, the change in Tz between two consecutive frames provides the cue for determining which one of the two possible yaw angles is correct, as illustrated in Figure 8, which is based on the following observation: If the yaw angle is greater than −90 (or smaller than 90 if the projectile is heading to the right), then the projectile is flying *away from* the viewing camera, resulting in a distance that increases with time. On the other hand, if the yaw angle is less than −90 (or greater than 90 if heading to the right), then the projectile is flying *toward* the viewing camera, yielding a distance that decreases with time. Therefore, by looking at the *sign of the rate of change in the distance dTz*, IMADE can determine if the yaw is greater than –90 (or less than 90 if heading to the right) or less than −90 (or greater than 90 if heading to the right). This yaw orientation strategy is used to obtain the results provided in Section 5 when *Tz* is known. When *Tz* is unknown, IMADE cannot use the *dTz* value from the estimated *Tz*, as it is highly non-smooth.

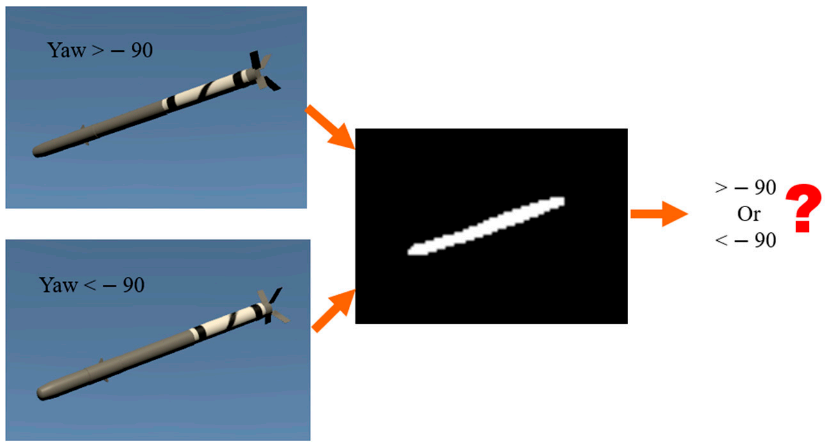


Figure 7. Ambiguity of yaw estimation with unknown *Tz*.

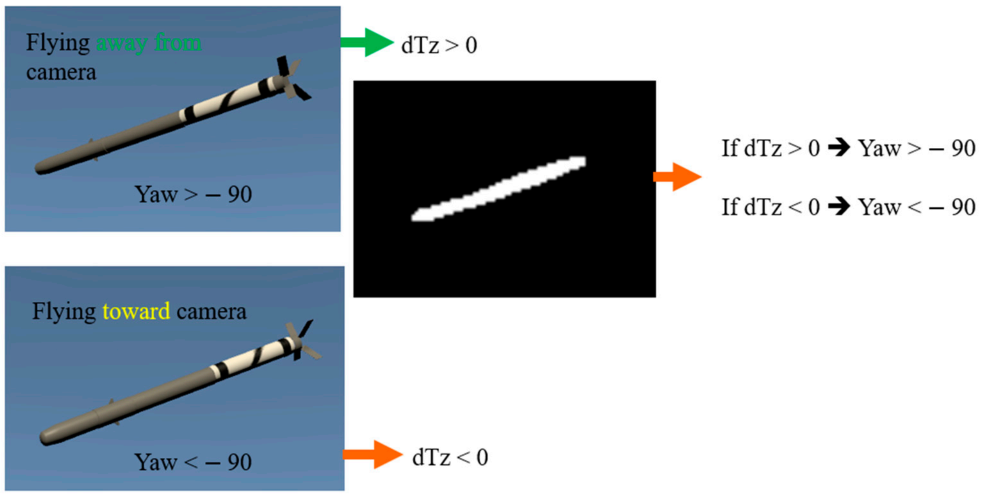


Figure 8. A solution for resolving the ambiguity of yaw estimation.

Estimation of Tx, Ty, and Tz

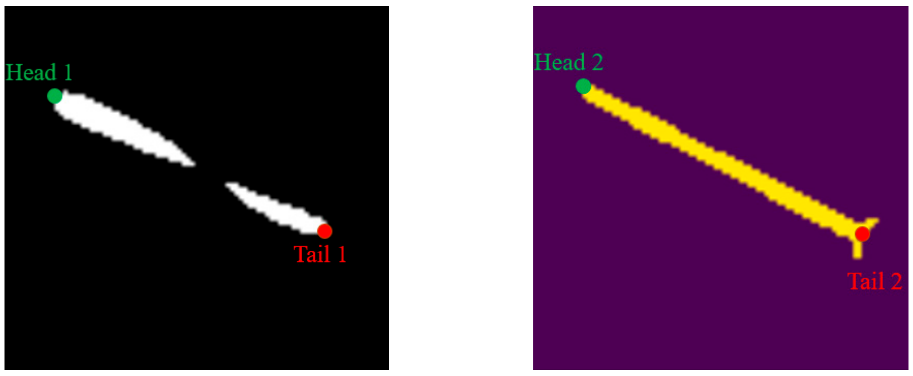
Tx, *Ty*, and *Tz* are the relative translations between the projectile and the viewing camera in the horizontal, vertical, and depth directions, respectively. *Tx* and *Ty* can be estimated from the centroid of the mask, as indicated in Figure 6a. *Tz*, the distance from the viewing camera to the target projectile, can usually be precisely measured with an additional range sensor and provided as prior knowledge, in addition to the exact 3D model of the projectile. Nevertheless, three approaches for the initial estimation of *Tz* are developed in this effort. The first approach is when the projectile center coincides with the image center. In this case, the pitch can be directly measured as the angle between the mask and the horizontal axis (see Figure 6a). Given the pitch angle, together with the measured height of the bounding box *H*, *L* can be computed accordingly and, in turn, can be exploited for the estimation of *Tz*, which is inversely proportional to *L*. The second approach for the estimation of *Tz* is based on the measured projectile diameter *d*, as indicated in Figure 6a. Unlike the projectile length, which cannot be measured directly from the mask, as both pitch and yaw are unknown, the projectile diameter is directly measurable. Based on the measured diameter d, together with the actual diameter of the projectile and intrinsic parameters of the camera, Tz can be directly inferred. Since the measured projectile diameter is obtained in the units of pixels, the estimated value of Tz will be discrete. See Figure 15c for an example. The third approach is carried out by matching the rendered projectile silhouette and the reference mask with respect to the pose parameters, including Tz.

4.2.2. Pose Estimation through Profile Matching

After image-/mask-based pose estimation, the estimated pose is fed to the second component consisting of differentiable rendering followed by optimization with respect to the pose parameters. There are three inputs: the reference mask, the initial pose estimate obtained using the mask-based method that was previously described, and the 3D model of the projectile. Differentiable rendering is used to render the projectile profile according to the given projectile pose estimate. Then, optimization is performed to minimize the difference between the rendered projectile profile and the given reference mask. In this work, two loss functions are considered. The first one, $loss_m$, is the mean squared error (MSE) between the reference image and the rendered projectile silhouette. Since the segmentation masks obtained from MaskR-CNN are not always single connected components (see Figure 9a for an example), $loss_m$ alone is not always sufficient. To overcome this issue of non-connected components, a second function, $loss_e$, is devised. We define $loss_e$ as the sum of the squared distances (SSD) between the positions of the head and tail points or *extreme points,* as depicted in this work; these are extracted from the reference image and the rendered silhouette. Figure 9 illustrates the extreme points extracted from a reference image and a rendered silhouette. Mathematically, the second loss function, $loss_e$, is defined as

$$loss_e = \text{dist}^2(Head1, Head2) + \text{dist}^2(Tail1, Tail2) \tag{4}$$

Then, the total loss is the simple weighted sum of $loss_m$ and $loss_e$.



(**a**) (**b**)

Figure 9. Extreme points in (**a**) a reference image, which is a mask detected with MaskR-CNN, and (**b**) a projectile profile rendered with Pytorch3D.

For differentiable rendering, we employed PyTorch3D [41], which provides efficient, reusable components for 3D computer vision research and works with PyTorch. When the distance Tz between the flying projectile and the viewing camera is known, the total loss is optimized with respect to all of the pose parameters except for Tz. On the other hand, when Tz is not known, all pose parameters are estimated through optimization. In order to obtain a smoother estimate of Tz, after profile matching is complete, we estimate Tz

as a linear combination of the current estimate of Tz and the previous estimate of Tz, as described in Equation (5):

$$\overline{T}_z(t) = r \cdot T_z(t) + (1 - r) \cdot T_z(t - 1) \tag{5}$$

where $\overline{T}_z(t)$ is the smoothed Tz at the current time t, $T_z(t)$ is the current estimate of Tz obtained from profile matching, $T_z(t - 1)$ is the estimate of Tz obtained at time $t - 1$, and r is a parameter used to control the smoothness of the resulting value of Tz.

4.3. Shape-Guided Segmentation

Three shape-guided segmentation approaches are devised and tested for IMADE. The first two are based on classical segmentation methods, and the third one is a machine learning (ML)-based method.

4.3.1. GVF-ASM

The GVF-ASM method incorporates Gradient Vector Flow Snakes [4] and active shape models [3]. Gradient Vector Flow (GVF) is an external force for Snakes that was proposed to increase the capture range and make it possible for a Snake to progress into prolonged concave regions. To devise a shape-guided segmentation method based on GVF, active shape models (ASMs) were considered. ASMs have been adopted for the incorporation of shape priors in various segmentation approaches [42,43] to overcome the common limitation of classical segmentation approaches due to their sensitivity to the quality of images. Active shape models are designed to capture the natural variability within a class of shapes for image searches in an iterative refinement algorithm analogous to that employed in active contour models. In other words, ASMs place strong constraints on how an active contour can evolve. Therefore, it is natural to combine ASMs and active-contour models in order to locate partially occluded objects in noisy, cluttered images.

The first step in an ASM is to construct a “point distribution model” (PDM) based on a set of examples of an object. The PDM is constructed by manually placing a set of points representing the structure of the object of interest. The set of points are then automatically aligned to minimize the variance in distance between equivalent points. The model gives the average positions of the points and has several parameters that control the main modes of variation found in the training set. Given such a model and a contaminated image containing an example of the object modeled, image segmentation involves choosing values for each of the parameters to find the best fit of the model to the image. To determine the best fit for each point, the ASM determines a displacement that moves it to a better location. In [3], a heuristic approach was used to calculate a suggested movement for each model point. Also suggested in [3] was the use of an external force field in the traditional active-contour model formulation. This is where GVF can be applied, as GVF was developed as an improved external force based on a given image. These local deformations are then transformed into adjustments for the pose, scale, and shape parameters of the PDM.

For IMADE, we developed and implemented a simplified active shape model, as described in Algorithm 1.

Algorithm 1: GVF-ASM

```
Input: Target chip and a shape prior in the form of a binary mask.
Output: Segmentation mask

Step-1: Set $i = 0$ and compute the gradient vector field of the target chip. Denote the current
        boundary points of the shape prior as $X_i$ and the gradient vector flow field there as $dX_i$.
Step-2: Find a rigid transformation $M(s,\theta)$ and $t = [t_x, t_y]^T$ such that $M(s,\theta) \cdot X_i$ + t is as close to $X_i$
        + $dX_i$ as possible, where $s$ is the scaling factor, $\theta$ is the rotation angle, and $t$ is the
        translational vector (we adopted the algorithm described in [3] for this step).
Step-3: Use the pose parameters $s$, $\theta$, $t_x$, and $t_y$ to transform $X_i$ and obtain the boundary points of
        the next iteration $X_{i+1}$.
```

4.3.2. SIS-Cut

The second shape-guided segmentation method devised was inspired by GraphCut segmentation. GraphCut is a combinatorial optimization method that can be used to find the global optimum of a pseudo-Boolean function f in polynomial time. In image segmentation applications, the pseudo-Boolean function f or, more commonly, the *energy function* E_i has the following form:

$$E_i = \mu \sum_{p \in \mathcal{P}} R_p(A_p) + \sum_{(p,q) \in \mathcal{N}: A_p \neq A_q} B_{p,q} \tag{6}$$

where p is a pixel, $\mathcal{P}$ is the set of all pixels constituting the image, A_p, which is 0 or 1, is the label for p, $R_p(A_p)$ is the individual pixel-matching cost (regional term) for pixel p, $\mathcal{N}$ is the set of neighboring pixels, $B_{p,q}$ is a boundary term that discourages intensity difference between p and q, and μ is a weight used to balance the regional term and the boundary term. GraphCut segmentation divides an input image into two regions, the background region R_B and the foreground region R_F. The division is approached such that (1) the intensity distributions are close to the known sample background and foreground distributions and (2) the intensity values of neighboring pixels belonging to different regions are as distinct as possible. GraphCut segmentation is not fully automatic in general, as it requires user input to estimate the required background and foreground distributions. IMADE incorporates a given shape prior into it for segmentation against highly cluttered backgrounds. To this end, inspired by the GraphCut segmentation criteria (1) and (2), we devised a procedure, Shape-Incorporated Segmentation with the GraphCut criteria (SIS-Cut), to maximize the distance between the distributions of the foreground and background regions defined by a segmentation mask. The details of SIS-Cut are given in Algorithm 2. In Algorithm 2, the transformation considered is 2D affine transformation with seven parameters: the scale in X, scale in Y, rotation angle, shear in X, shear in Y, translation in X, and translation in Y. Also, *Nelder–Mead* simplex optimization is employed so that gradient information is not required in the optimization process.

```
Algorithm 2: SIS-Cut

Input: Target chip and a shape prior in the form of a binary mask.
Output: Segmentation mask

Step-1: Denote the region covered by the given shape prior as $\mathbb{P}$. Perform morphological dilation
        to expand the covered region and denote it as $\mathbb{P}'$.
Step-2: Set $\mathbb{P}$ as the foreground region and $\mathbb{P}' \backslash \mathbb{P}$ as the background region. Obtain the probability
        mass functions of the intensities of the two regions and denote them as $p_F$ and $p_B$,
        respectively.
Step-3: Employ the Nelder–Mead simplex optimization technique to maximize the following
        criterion with respect to the 2D transformation parameters:
            $dist(p_F,p_B) = \frac{1}{2}(KL(p_F, p_B) + KL(p_B, p_F))$ (7)
```

4.3.3. 2D TETRIS

TETRIS [6] stands for Template Transformer Networks for Image Segmentation with Shape Priors, and it was originally described for the segmentation of coronary lumen structures in 3D cardiac computed tomography. The novelty of TETRIS lies in the introduction of template transformer networks, where a shape template is deformed to match the underlying structure of interest through an end-to-end trained spatial transformer network [44]. TETRIS has the advantage of explicitly enforcing shape priors without the need to devise an image-to-segmentation likelihood function. This is achieved by leveraging the representational power of neural networks while explicitly enforcing shape constraints in segmentations by restricting the model to perform segmentation through deformations of a given shape prior. Additionally, TETRIS guarantees that the segmentation is a single connected component, as it does not restrict the network to making pixel-wise classifications. Instead, a neural network (NN) is trained to align the shape prior to the structure of interest that is visible in the input image. Using priors is especially advantageous when the data are ambiguous, such as those with a low resolution, poor target contrast, partial occlusion, and a cluttered background. The workflow of 2D TETRIS is given in Figure 10. The input of the network consists of the target chip that contains the flying projectile and a given shape prior in the form of a binary template, as shown in the purple dashed box. The network consists of two modules: a transformation prediction module and a template warping module. The output is a warped template serving as the segmentation of the flying projectile.

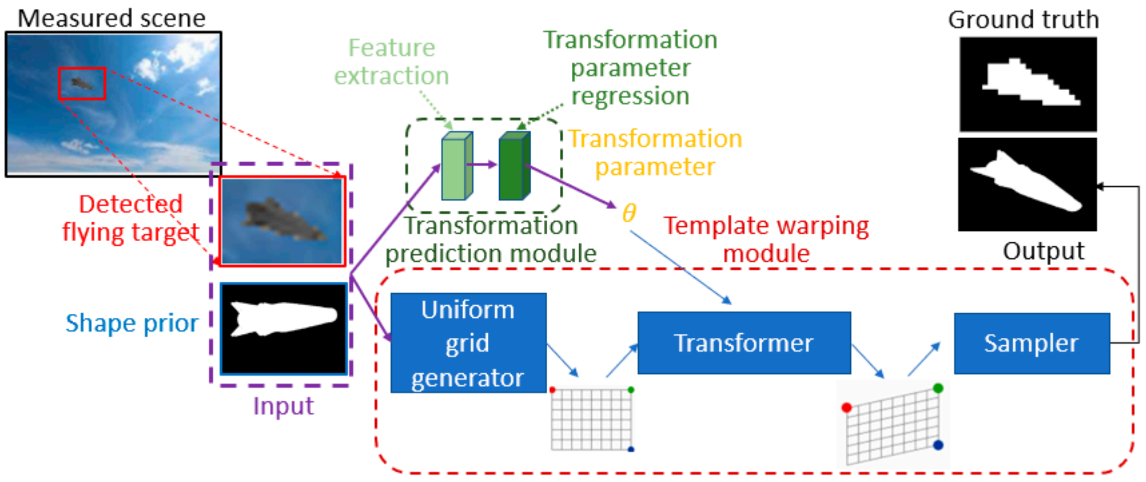


Figure 10. Workflow of the proposed 2D template transformer network-based segmentation.

Transformation Prediction Module

There are two parts of the transformation prediction (TP) module. The first one is a feature extractor, and the second one is a transformation parameter regressor. Regarding the feature extractor, most of the existing feature extraction neural networks used for object classification tasks can be employed, e.g., Visual Geometry Group (VGG) [45], ResNet [46], and different versions of Inception networks [47–49]. The transformation parameter regressor (TPR) developed in [6] was complicated, since it adopted a model that was free from deformation. The TPR model requires deformation field regularization to avoid physically non-plausible deformation. In the IMADE application, 2D rigid or affine transformations involving four to six parameters suffice. For this reason, 1–2 simple fully connected layers are sufficient to predict the required transformation parameters.

Template Warping Module

The purpose of the template warping module is to generate a warped template based on the transformation parameters received from the transformation prediction module. Three steps are involved in this module. First, the uniform grid generator generates a uniform grid based on the size of the image patch containing the detected flying target. For example, if the size of the image patch is $m \times n$, then a uniform grid of $m \times n$ will be generated in this step. Next, the transformer in the second step is used to generate a dense deformation field based on the transformation parameters received from the first module. The result is a deformed grid, as illustrated in Figure 10. The final step includes the sampler, which samples the input shape prior (the template, a binary mask) to generate the warped template, as shown in Figure 10. During training, the warped template is compared with the ground-truth segmentation to compute the loss function to be minimized. A simple L_2 norm loss function suffices. Since the loss is differentiable with respect to the transformation parameters, the complete network can be trained end to end.

Shape Prior Generation

One important step in training 2D TETRIS is the generation of shape priors. For training purposes, the shape priors can be generated based on the ground-truth segmentation mask. By randomly and slightly altering each ground-truth pose parameter, many shape priors can be generated for each target chip. In our experiments, which are provided in Section 5.4, five shape priors were generated for each target chip based on the ground-truth segmentation masks for the training set. An alternate and potentially more effective strategy is to randomly generate training shape priors at runtime. We envision that this strategy will eliminate the potential overfitting problem, as the shape priors are not pre-generated.

5. Experimental Procedures and Results

This section first describes the datasets used in the experiments, followed by the procedures and results obtained from each module depicted in Figure 1.

5.1. Datasets

Both simulated and real data are considered in our experiments. Simulated data allow us to quantitatively analyze the results, while only qualitative analysis is available when real data are employed.

5.1.1. Simulated Data

Figure 11 provides the scene setting used to generate the datasets for the experiments, including the background terrain and the locations of the projectile launcher and target in Unity's world coordinate frame in terms of meters. Three video clips were generated: training, validation, and test clips. Table 1 provides the configurations adopted in each clip. In Table 1, we see that the three datasets were generated with different projectile launch angles, times of day, and random seeds, which ensured that no identical target chips were used in the training, validation, and testing phases. Three simulated target chips are provided in Figure 2d.

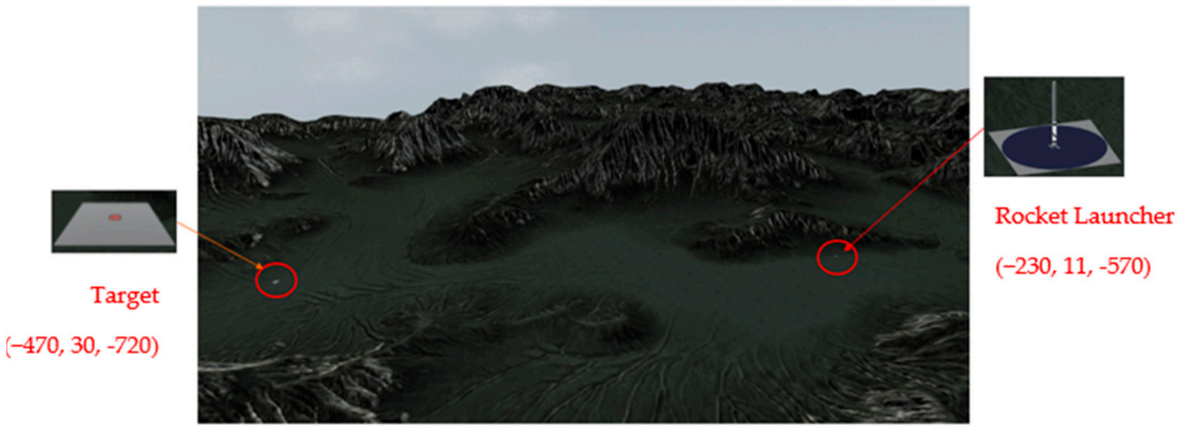


Figure 11. Scene setting used to generate the simulation data.

Table 1. The datasets generated by the IMADE simulation tool.

	Train	Validation	Test
Launch Angle	40	41	39
Time of the day	17:00	17:30	17:15
Random seed	539662031	539662032	539662033
Weather	Cloudy	Cloudy	Cloudy
Camera	1	1	1
# of frames	175	175	175

5.1.2. Real Data

We qualitatively tested our solution on one real video clip. Several sample target chips with a size of 256 × 256 that were extracted from it are provided in Figure 12. The first 363 frames were purposely skipped, as the targets could not be reliably captured around the center of the images within this range. In Figure 12, we observe that the projectile's visibility worsens with the increase in its frame number due to the increasing distance from the viewing camera.

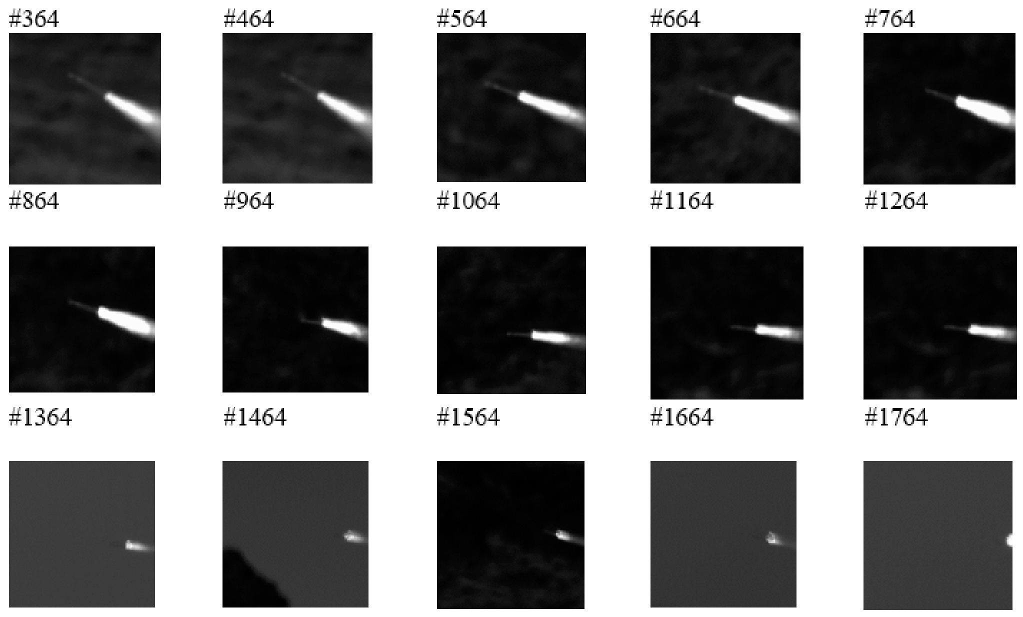


Figure 12. Example target chips cropped from the center of the real flying projectile video.

5.2. Mask R-CNN

5.2.1. Simulated Data

We employed the pre-trained mask-rcnn-R-50-FPN-3x model from Detectron2 [50]. This pre-trained model was fine-tuned by using 175 target chips cropped from the center of the training dataset. The Mask R-CNN module was configured such that only the detection with the top confidence score was returned. Several exemplary MaskR-CNN detection results together with the ground-truth segmentation masks in red are provided in Figure 13.

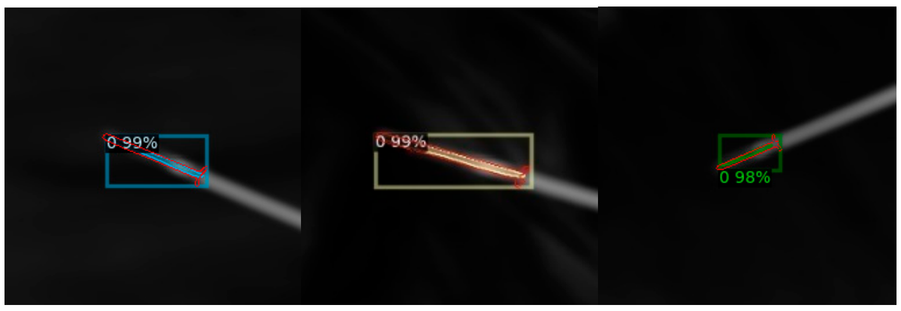


Figure 13. Exemplary Mask-R-CNN detection results (simulation data).

5.2.2. Real Data

The same fine-tuned MaskR-CNN model was used in this experiment. However, when Mask R-CNN was configured to return only the masks with the top confidences for the real dataset, we noticed that it sometimes wrongly detected the projectile exhaust as the projectile. We also noticed that it may have included the projectile exhaust in the returned mask. However, if it was configured to return the top two masks, then the correct mask was almost always included. For this reason, when we applied Mask R-CNN on the real dataset, it was configured to detect the masks with the top two confidence scores. Then, by computing the mean intensity of the regions covered by the masks and selecting the one with a lower value to exclude the possible projectile exhaust, the correct mask could be detected and selected. Typical results are provided in Figure 14, where we observe that it worked reasonably well.

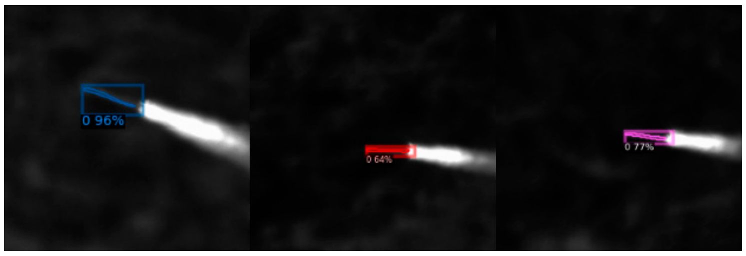


Figure 14. Exemplary Mask-R-CNN detection results (real data).

5.3. Mask-Based Pose Estimation

Once the initial target segmentation mask was obtained from MaskR-CNN, the next step was to apply the mask-based pose estimation approach described in Section 4.2.1 to estimate the projectile pose. We note that profile-matching-based pose estimation was not employed when the segmentation mask was obtained from MaskR-CNN. This was due to the fact that MaskR-CNN produced segmentation masks that were not shape-preserving, making the profile-matching-based approach either problematic or sub-optimal.

5.3.1. Simulated Data

When simulation data are used, since the ground-truth annotation is available, the cases of both unknown Tz and known Tz are considered.

Unknown Tz

When Tz is not known, we estimate it by measuring the diameter d of the projectile from each 2D image frame, as described in Section 4.2.1. The estimated translations and pitch and yaw angles are provided in Figure 15, where the units of Tx and Ty are in pixels, Tz is in meters, and the pitch and yaw are in degrees. As discussed in Section 4.2.1, since the measured projectile diameter d is discrete, so is the estimated distance Tz, as observed in Figure 15. The ambiguity of yaw estimation is clearly observed.

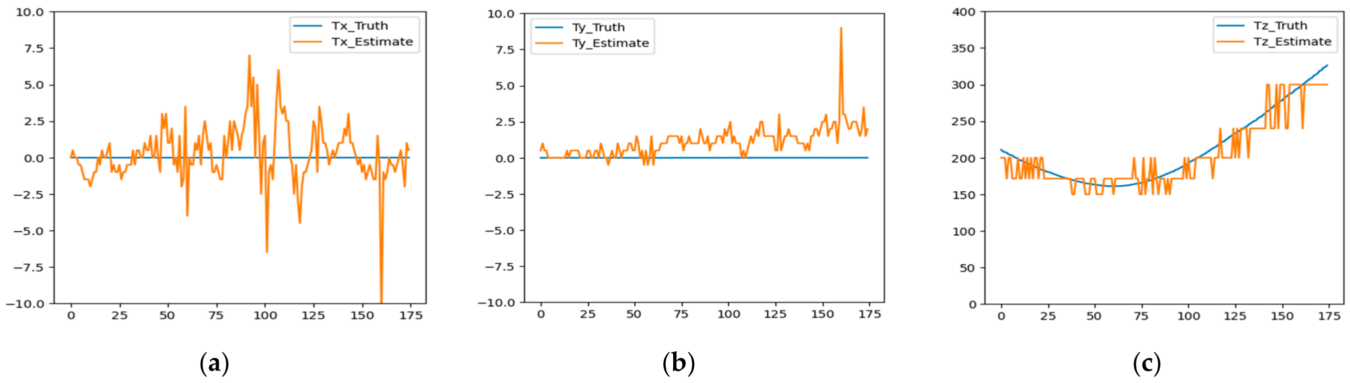
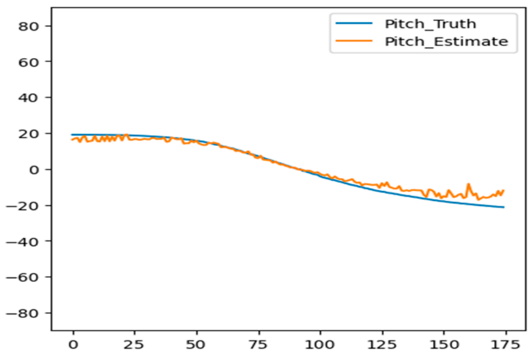
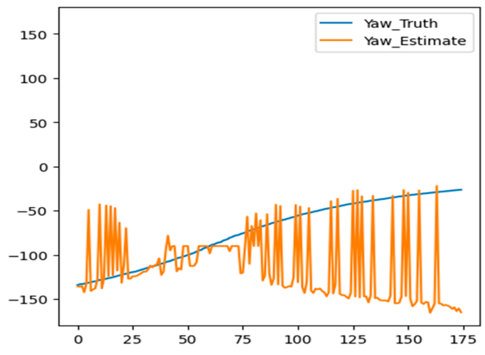


Figure 15. *Cont.*



(**d**)



(**e**)

Figure 15. Estimated pose parameters (orange) and their ground-truth values (blue). From left to right: (**a**) horizontal displacement in pixels, (**b**) vertical displacement in pixels, (**c**) distance in meters, (**d**) pitch angle in degrees, and (**e**) yaw angle in degrees.

Known Tz

When Tz is assumed to be known, the ambiguity of yaw estimation is resolved by referencing the sign of dTz, as discussed in Section 4.2.1. The estimated in-plane translations, pitch, and yaw are given in Figure 16, where the in-plane translations are exactly the same as those obtained previously, as they do not depend on Tz. However, a slight improvement in the pitch estimate is observed, and the ambiguity of yaw estimation is resolved.

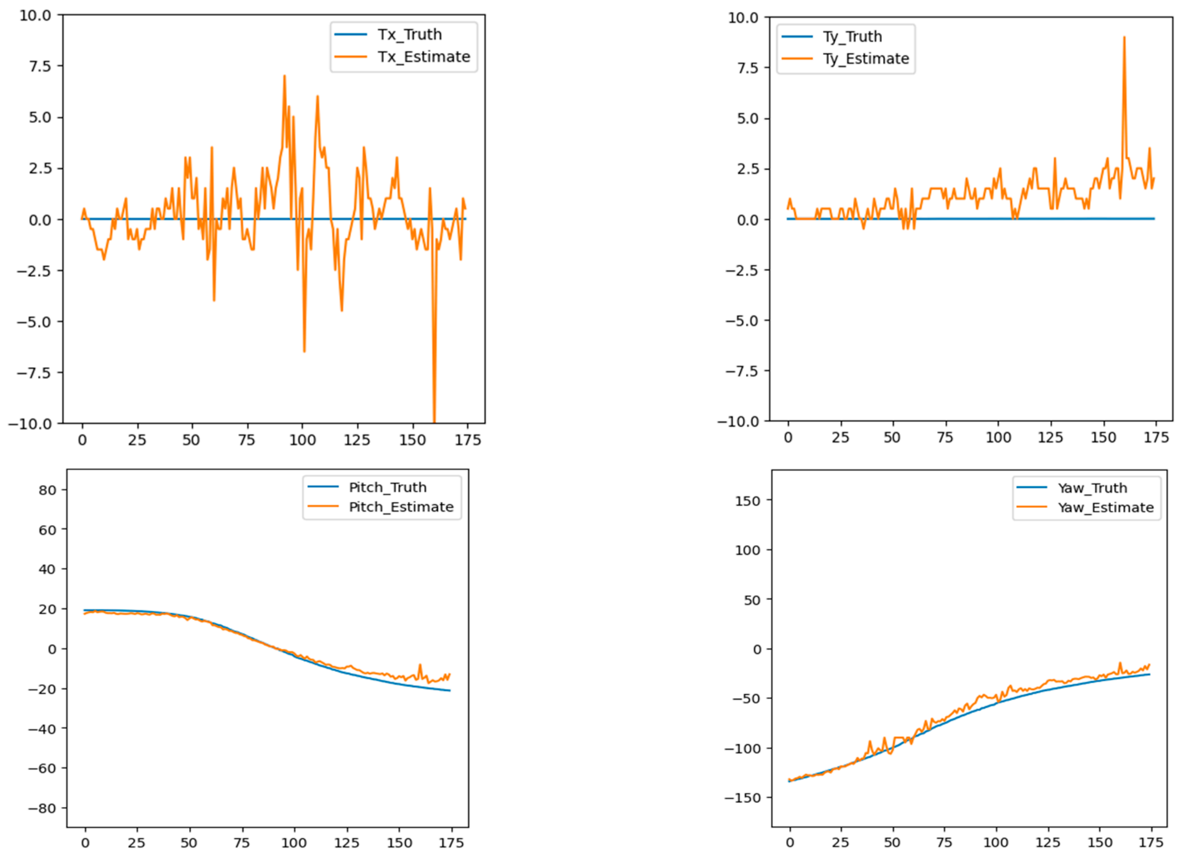


Figure 16. Estimated pose parameters (orange) and their ground-truth values (blue) when Tz is known. From left to right: horizontal displacement in pixels, vertical displacement in pixels, pitch angle in degrees, and yaw angle in degrees.

5.3.2. Real Data

For the real dataset, since no ground truth is available, we present the results in terms of the generated shape priors.

In each image shown in Figure 17, the mask generated with MaskR-CNN is displayed in green, and the shape prior generated by rendering the projectile silhouette from the 3D projectile model using the estimated pose parameters is displayed in red. In Figure 17, we observe that the estimated target poses are in good agreement with the masks detected with MaskR-CNN, which validates the mask-based pose estimation approach described in Section 4.2.1, but it also indicates that the performance of MaskR-CNN is crucial, as it directly impacts the accuracy of the determined target attitudes.

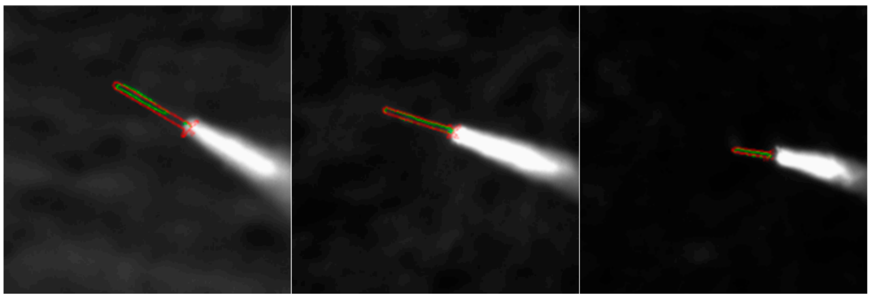


Figure 17. Masks detected with MaskR-CNN (green) and the associated shape priors (red) generated from the pose parameters estimated by using the mask-based pose estimation approach.

5.4. Shape-Guided Segmentation

We present the quantitative segmentation results using the Jaccard index or intersection over union (IoU) score when simulated data are used, and the qualitative results are presented by displaying segmentation masks on top of the target chips when the real data are used in this section. Two cases are considered when simulated data are used, depending on if the initial pose parameters are estimated with unknown Tz or known Tz.

5.4.1. Simulated Data

We first provide a visual comparison among the different segmentation methods by displaying the segmentation masks in a typical target chip in Figure 18, where the detection with MaskR-CNN detection, the shape prior generated from the estimated pose, and the segmentation masks obtained from the three shape-guided segmentation approaches are displayed in yellow and are superimposed with the ground-truth masks in green. Visually, we observe that MaskR-CNN performs very well, except that the detailed shape of the projectile tail is not preserved. Also observed is that 2D TETRIS outperforms GVF-ASM and SIS-Cut. To evaluate them quantitatively, Figure 19 presents the IoU scores. On each box, the central mark indicates the median, and the bottom and top edges of the box indicate the 25th and 75th percentiles, respectively. Figure 19 shows the IoU scores of different shape-guided segmentation approaches. (a) Shape priors are estimated while assuming unknown Tz. (b) Shape priors are estimated while assuming known Tz.

Whisker plots were created to extend to the most extreme data points that were not considered outliers, and the outliers were plotted individually using the '+' marker symbol. Several observations can be made from Figure 19. (1) MaskR-CNN performs very well on this simulation dataset, with a median IoU score of 0.64, considering that it is a blind detection and segmentation method that outputs rough profiles of each target, which explains why the best IoU scores are under 0.8. (2) The shape priors generated according to the pose estimated with known Tz are slightly better than those generated with unknown Tz in terms of the median IoU score. (3) Though the best IoU scores of the generated shape priors are better than those of MaskR-CNN detection in both cases, their median values are lower. A possible reason is that the roll angle is not estimated in this work, which results in a mismatch in the tail of the projectile when the shape prior is rendered from the

3D model. See Figure 20 for more details. (4) Both GVF-ASM and SIS-Cut are unable to improve the segmentation accuracy. In fact, the resulting segmentation masks are worse than the given shape priors estimated from the MaskR-CNN detection. (5) The machine-learning-based approach, 2D TETRIS, however, significantly improves the segmentation accuracy by increasing the median IoU scores by about 30% (from about 0.6 to 0.78) and greatly reduces the difference in the 25th and 75th percentiles of the IoU scores in the cases of both unknown *Tz* and known *Tz*.

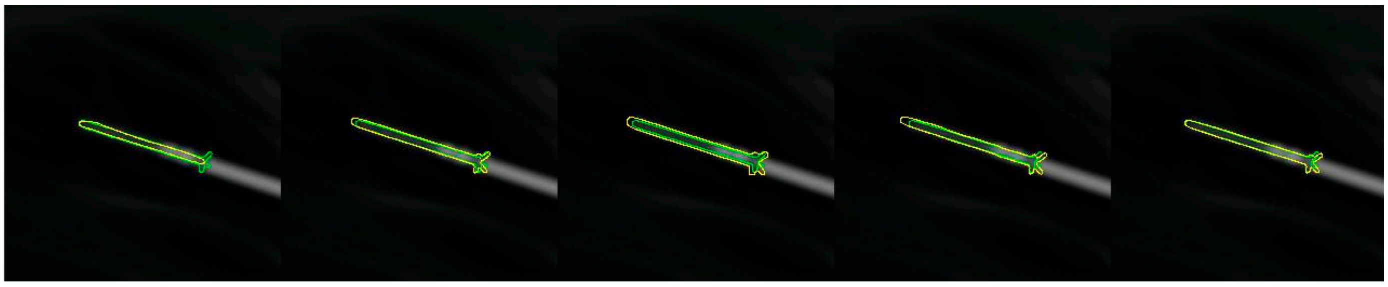
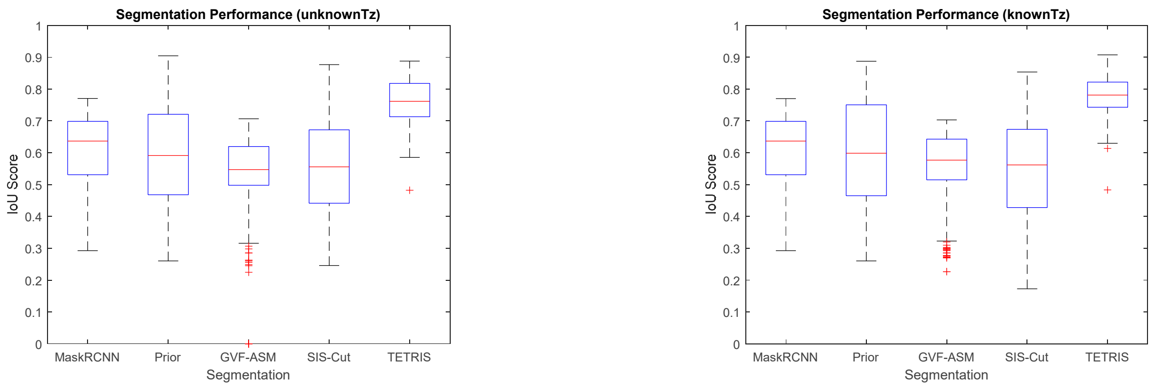

(**a**) (**b**) (**c**) (**d**) (**e**)

Figure 18. Qualitative comparison of different segmentation approaches. (**a**) MaskR-CNN; (**b**) generation of the shape prior according to the estimated pose; (**c**) GVF-ASM segmentation; (**d**) SIS-Cut segmentation; (**e**) 2D TETRIS.



(**a**) (**b**)

Figure 19. IoU scores of different shape-guided segmentation approaches. (**a**) Shape priors are estimated while assuming unknown *Tz*. (**b**) Shape priors are estimated with known Tz.

(**a**) (**b**)

Figure 20. An example showing why the IoU scores of MaskR-CNN detection can be higher than those of the corresponding shape priors. (**a**) MaskR-CNN detection (yellow) and ground-truth segmentation (green) (**b**) The corresponding shape prior (yellow) and ground-truth segmentation (green). Notice that the MaskR-CNN detection mask is completely inside the ground-truth segmentation mask, while the shape prior rendered with a 3D model has a visible difference at the tail of the projectile due to the mismatched roll angle, which explains why the median IoU score of the shape prior is lower than that of MaskR-CNN detection.

5.4.2. Real Data

We used the same MaskR-CNN model in this experiment. Since no ground truth was available when a real video clip was used, in Figure 21, we provide the segmentation results for the 364th, 564th, and 764th frames in red and the given shape priors in yellow for MaskR-CNN detection and the three shape-guided segmentation methods. Surprisingly, the MaskR-CNN model that was fine-tuned with synthetic data performed quite well on this real video clip, and visually, 2D TETRIS performed better than the GVF-ASM and SIS-Cut methods.

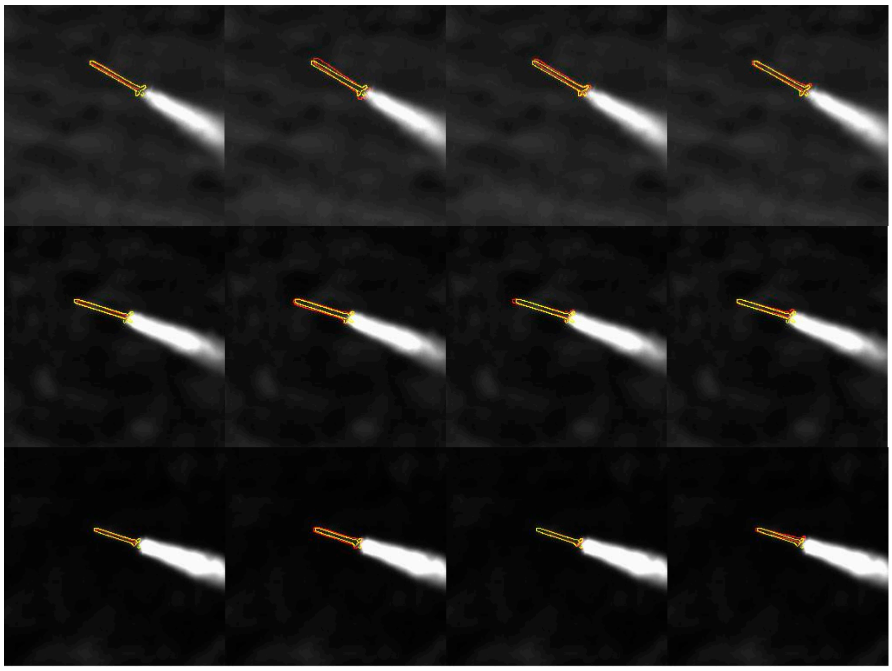


Figure 21. Real data segmentation results (red) and associated given shape priors (yellow) in the real dataset. From left to right: Mask R-CNN, GVF-ASM, SIS-Cut, and 2D TETRIS.

5.5. Performance Evaluation

This section compares the accuracies of the projectile attitudes estimated from the segmentation masks resulting from MaskR-CNN, GVF-ASM, SIS-Cut, and 2D TETRIS when using the simulation data with ground-truth annotations. Unlike the results presented in Section 5.3.1, when the segmentation masks were obtained from the shape-guided segmentation module, three pose estimation methods were employed. The pose estimation methods were mask-based estimation, pose estimation via profile matching without Tz smoothing, and pose estimation via profile matching with Tz smoothing. In the last case of profile matching, the parameter r in Equation (5) was set to 0.1.

5.5.1. Unknown Tz

Five pose parameters—Tx, Ty, Tz, pitch, and yaw—are estimated when the distance Tz is unknown. Shown in Figure 22 are the accuracies of each pose parameter, except for yaw, normalized to a distance equal to one kilometer. Yaw estimates are not included due to the ambiguity of yaw estimation when Tz is unknown. Three pose estimation methods are used for each segmentation method, except for MaskR-CNN. This is because segmentation with MaskR-CNN is not shape-preserving, making profile matching problematic. Overall, the Tz and pitch estimation results support that 2D TETRIS segmentation results in the best pose estimation.

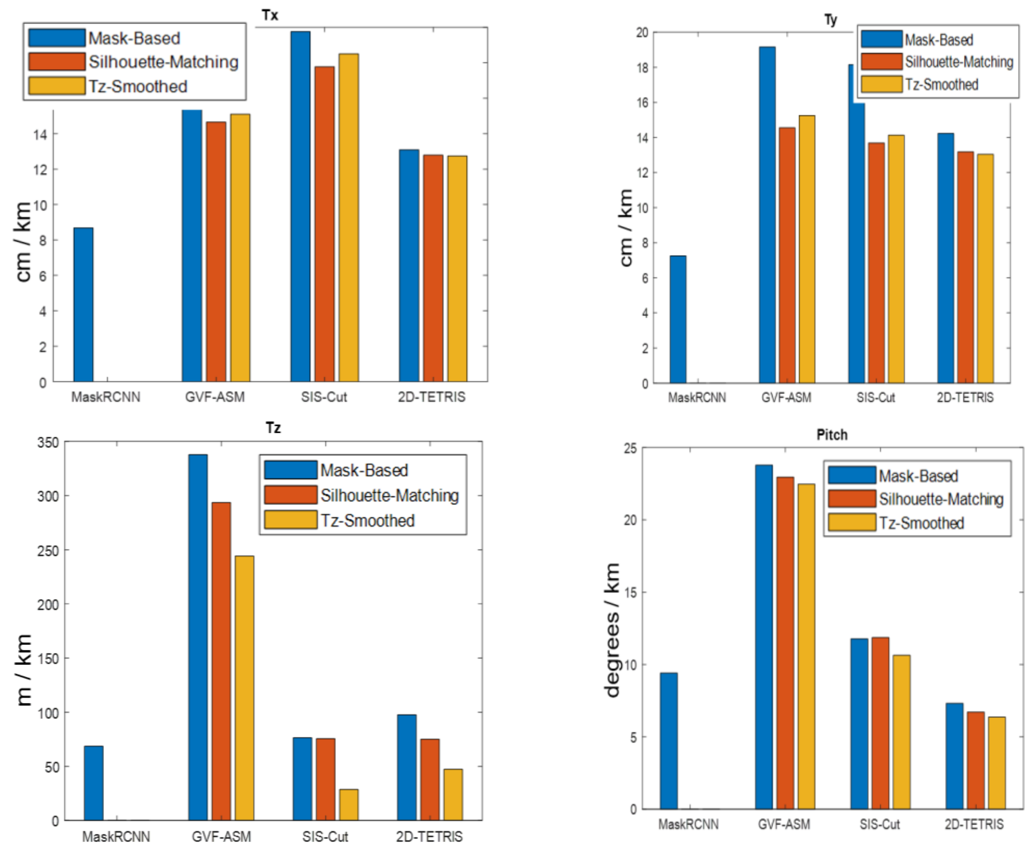


Figure 22. Comparison of the projectile attitude determination performance *in the case of unknown Tz* among four different segmentation approaches: detection and segmentation with MaskR-CNN, GVF-ASM, SIS-Cut, and 2D TETRIS. Except for MaskR-CNN, three pose estimation approaches are used to estimate each pose parameter.

5.5.2. Known Tz

When *Tz* is known, four pose parameters—*Tx*, *Ty*, pitch, and yaw—are estimated. Shown in Figure 23 are the accuracies normalized to a distance equal to one kilometer. Similarly to the case of unknown distance, the 2D TETRIS segmentation results in the best performance, as observed from the pitch and yaw estimation results. Specifically, the estimation errors for pitch and yaw are within 3 degrees/km and 20 degrees/km when 2D TETRIS is employed using the *Tz*-smoothed pose estimation approach. For the horizontal and vertical distance with respect to the viewing camera, all methods result in an error of less than 0.2 m/km, and again, MaskR-CNN results in the most accurate target location estimations. In addition, regarding the same segmentation and pose estimation methods, the accuracies achieved when *Tz* is known are higher than those achieved when *Tz* is unknown.

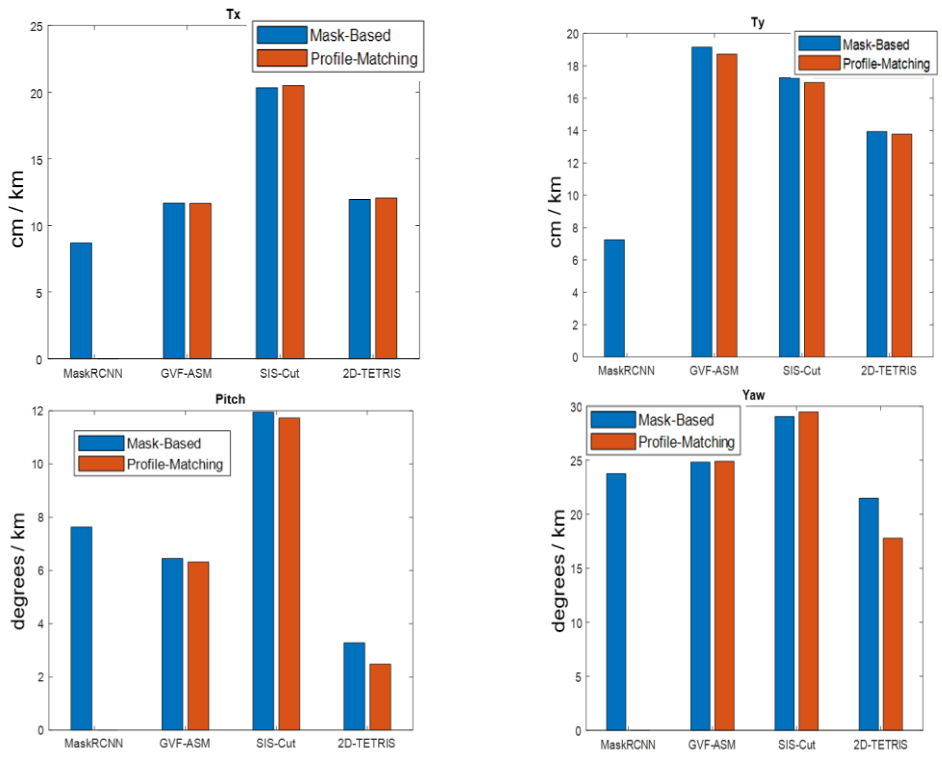


Figure 23. Comparison of the projectile attitude determination performance *in the case of known Tz* among four different segmentation approaches: detection and segmentation with MaskR-CNN, GVF-ASM, SIS-Cut, and 2D TETRIS. Except for MaskR-CNN, three pose estimation approaches are used to estimate each pose parameter.

5.5.3. Runtime Comparison

Lastly, we report the actual runtime performance of each of the three shape-incorporated segmentation approaches considered in our experiments. In Table 2, we detail the processing time for each segmentation method when handling 175 simulated images with a size of 256 × 256. These experiments were conducted on a PC equipped with an Intel i5-9400 CPU running at 2.90 GHz with 16 GB RAM and an NVIDIA GeForce RTX 3060 graphics card with 12 GB memory.

Table 2. Actual runtimes of the three considered shape-incorporated segmentation approaches.

	GVF-ASM	**GraphCut Criterion-Based**	**2D TETRIS**
Time	1502 (s)	995 (s)	**7.12 (s)**
Framerate (ratio)	~0.12 fps (1)	~0.18 fps (1.5)	~25 fps (208)
Code	Matlab^{TM}	Matlab^{TM}	Python
OS	Windows 10	Windows 10	Ubuntu 20.04
Hardware	CPU	CPU	CPU + GPU

Notably, the machine-learning-based method, 2D TETRIS, demonstrates remarkable efficiency and holds the potential to result in a near real-time application with proper software and hardware optimization.

6. Conclusions

This paper presents a model-based Imagery Projectile Attitude Detection and Estimation (IMADE) method for ground-based monocular imagery. Due to the limitations of the imagery resolution, signal-to-noise ratio, target contrast, and potentially highly cluttered background, IMADE performs image segmentation and estimates the projectile attitude by matching a segmentation mask and a profile rendered from its 3D model. IMADE consists of three modules: a detection and segmentation module, a profile-matching module, and a shape-guided segmentation module. For the segmentation module, IMADE adopts a machine-learning-based method, Mask R-CNN, to simultaneously perform blind projectile detection and segmentation. A pre-trained model that was fine-tuned by using only 175 simulated target chips was shown to work very well on both simulated and real datasets. Next, in the profile-matching module, a two-stage approach was developed. In the first stage, IMADE directly estimates the attitude of the flying projectile from its segmentation mask, resulting in an attitude estimate, which initializes the second stage of estimation, where the projectile attitude is estimated by matching the segmentation mask and its projectile profile generated with a 3D model. Finally, the projectile profile generated with the 3D model is used as a shape prior in the last shape-guided segmentation module to improve projectile segmentation accuracy. The resulting segmentation mask is then fed back to the profile-matching module again to refine the projectile attitude estimate. Three shape-guided segmentation methods are devised and compared in this work. The first two, GVF-ASM and SIS-Cut, are related to classical active-contour segmentation and GraphCut segmentation, and the third one, 2D TETRIS, is a machine-learning-based method. Our preliminary experiments involving a simulated video clip with 175 frames and a real video clip with 600 frames indicate that the machine-learning-based approach, *2D TETRIS*, outperforms the two segmentation methods that are based on classical segmentation approaches. Specifically, our results from the simulated dataset indicate that, when the range information can be reliably obtained and segmentation with 2D TETRIS is adopted, the projectile attitude can be determined within 0.2 m in translation, 3 degrees in pitch angle, and 20 degrees in yaw angle when the target–camera distance is normalized to one kilometer.

Future directions that can be pursued to improve the projectile attitude determination accuracy include the following: (1) Instead of using single ground-based monocular images, images collected from multiple ground-based monocular cameras can be used. Multiple perspectives potentially eliminate the ambiguity of yaw estimation discussed in this paper by imposing consistent estimates from multiple cameras [46] and increasing the estimation accuracy. (2) Instead of the offline shape prior generation used in the training phase of 2D TETRIS, an online shape prior generation approach can be adopted to dramatically increase the diversity of training shape priors. (3) Temporal information can be more effectively exploited to improve the performance. Currently, the temporal information is only used to resolve the ambiguity of yaw estimation when Tz is known and to smooth the Tz estimate when it is not known a priori, yet it is shown to improve the accuracies of all pose parameters. We plan to investigate how to exploit the temporal information to detect outliers in the results of MaskR-CNN, which are the basis for accurate attitude determination.

Author Contributions: Conceptualization, H.C., Z.Z. and H.T.; methodology, H.C.; software, H.C., Z.Z. and H.T.; validation, H.C., Z.Z. and H.T.; formal analysis, H.C., Z.Z. and H.T.; investigation, H.C.; resources, E.B.; data curation, Z.Z. and H.T.; writing—original draft preparation, H.C.; writing—review and editing, E.B. and K.D.P.; visualization, H.C.; supervision, G.C.; project administration, G.C.; funding acquisition, G.C. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: It does not require ethical approval.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are available upon request from the corresponding author.

Conflicts of Interest: Author Huamei Chen was employed by the company Intelligent Fusion Technology. The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

References

1. Hinterstoisser, S.; Holzer, S.; Cagniart, C.; Ilic, S.; Konolige, K.; Navab, N.; Lepetit, V. Multimodal templates for real-time detection of texture-less objects in heavily cluttered scenes. In Proceedings of the 2011 International Conference on Computer Vision (ICCV), Barcelona, Spain, 6–13 November 2011; IEEE: Piscataway, NJ, USA, 2011.
2. Cao, Z.; Sheikh, Y.; Banerjee, N.K. Real-time scalable 6DOF pose estimation for textureless objects. In Proceedings of the 2016 IEEE International Conference on Robotics and Automation (ICRA), Stockholm, Sweden, 16–21 May 2016; IEEE: Piscataway, NJ, USA, 2016.
3. Cootes, T.F.; Taylor, C.J.; Cooper, D.H.; Graham, J. Active shape models-their training and application. *Comput. Vis. Image Underst.* **1995**, *61*, 38–59. [CrossRef]
4. Xu, C.; Prince, J.L. Gradient vector flow: A new external force for snakes. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, San Juan, PR, USA, 17–19 June 1997; pp. 66–71.
5. Boykov, Y.Y.; Jolly, M.-P. Interactive Graph Cuts for Optimal Bounday& Region Segmentation of Objects in N-D Images. In Proceedings of the Eighth IEEE International Conference on Computer Vision. ICCV 2001, Vancouver, BC, Canada, 7–14 July 2001.
6. Lee MC, H.; Petersen, K.; Pawlowski, N.; Glocker, B.; Schaap, M. Tetris: Template transformer networks for image segmentation with shape priors. *IEEE Trans. Med. Imaging* **2019**, *38*, 2596–2606. [PubMed]
7. Lee, A.; Dallmann, W.; Nykl, S.; Taylor, C.; Borghetti, B. Long-range pose estimation for aerial refueling approaches using deep neural networks. *J. Aerosp. Inf. Syst.* **2020**, *17*, 634–646. [CrossRef]
8. Wang, Y.; Wang, H.; Liu, B.; Liu, Y.; Wu, J.; Lu, Z. A visual navigation framework for the aerial recovery of UAVs. *IEEE Trans. Instrum. Meas.* **2021**, *70*, 1–13. [CrossRef]
9. Yang, J.; Wang, C.; Jiang, B.; Song, H.; Meng, Q. Visual perception enabled industry intelligence: State of the art, challenges and prospects. *IEEE Trans. Ind. Inform.* **2020**, *17*, 2204–2219. [CrossRef]
10. Opromolla, R.; Fasano, G.; Rufino, G.; Grassi, M. A review of cooperative and uncooperative spacecraft pose determination techniques for close-proximity operations. *Prog. Aerosp. Sci.* **2017**, *93*, 53–72. [CrossRef]
11. Sharma, S.; Park, T.H.; D'Amico, S. Spacecraft Pose Estimation Dataset (SPEED). Stanford Digital Repository. Available online: https://zenodo.org/records/6327547 (accessed on 21 December 2023).
12. Zhu, M.; Derpanis, K.G.; Yang, Y.; Brahmbhatt, S.; Zhang, M.; Phillips, C.; Lecce, M.; Daniilidis, K. Single image 3D object detection and pose estimation for grasping. In Proceedings of the 2014 IEEE International Conference on Robotics and Automation (ICRA), Hong Kong, China, 31 May–7 June 2014; IEEE: Piscataway, NJ, USA, 2014.
13. Marchand, E.; Hideaki, U.; Fabien, S. Pose estimation for augmented reality: A hands-on survey. *IEEE Trans. Vis. Comput. Graph.* **2015**, *22*, 2633–2651. [CrossRef] [PubMed]
14. Yu, Y.K.; Wong, K.H.; Chang, M. Pose estimation for augmented reality applications using genetic algorithm. *IEEE Trans. Syst. Man Cybern. Part B (Cybernetics)* **2005**, *35*, 1295–1301. [CrossRef] [PubMed]
15. Ferrara, P.; Piva, A.; Argenti, F.; Kusuno, J.; Niccolini, M.; Ragaglia, M.; Uccheddu, F. Wide-angle and long-range real time pose estimation: A comparison between monocular and stereo vision systems. *J. Vis. Commun. Image Represent.* **2017**, *48*, 159–168. [CrossRef]
16. Kehl, W.; Milletari, F.; Tombari, F.; Ilic, S.; Navab, N. Deep learning of local rgb-d patches for 3d object detection and 6d pose estimation. In Proceedings of the European Conference on Computer Vision, Amsterdam, The Netherlands, 11–14 October 2016; Springer: Cham, Switzerland, 2016.
17. Vidal, J.; Lin, C.-Y.; Marti, R. 6D pose estimation using an improved method based on point pair features. In Proceedings of the 2018 4th International Conference on Control, Automation and Robotics (Iccar), Auckland, New Zealand, 20–23 April 2018; IEEE: Piscataway, NJ, USA, 2018.
18. Gao, Y.; Dai, Q. View-based 3D object retrieval: Challenges and approaches. *IEEE Multimed.* **2014**, *21*, 52–57. [CrossRef]
19. Muja, M.; Rusu, R.B.; Bradski, G.; Lowe, D.G. Rein-a fast, robust, scalable recognition infrastructure. In Proceedings of the 2011 IEEE International Conference on Robotics and Automation, Shanghai, China, 9–13 May 2011; IEEE: Piscataway, NJ, USA, 2011.
20. Hinterstoisser, S.; Cagniart, C.; Ilic, S.; Sturm, P.; Navab, N.; Fua, P.; Lepetit, V. Gradient response maps for real-time detection of textureless objects. *IEEE Trans. Pattern Anal. Mach. Intell.* **2011**, *34*, 876–888. [CrossRef] [PubMed]
21. Gu, C.; Ren, X. Discriminative mixture-of-templates for viewpoint classification. In *European Conference on Computer Vision*; Springer: Berlin/Heidelberg, Germany, 2010.
22. Sundermeyer, M.; Marton, Z.C.; Durner, M.; Brucker, M.; Triebel, R. Implicit 3d orientation learning for 6d object detection from rgb images. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018.
23. Hodan, T.; Haluza, P.; Obdrzalek, S.; Matas, J.; Lourakis, M.; Zabulis, X. T-LESS: An RGB-D dataset for 6D pose estimation of texture-less objects. In Proceedings of the 2017 IEEE Winter Conference on Applications of Computer Vision (WACV), Santa Rosa, CA, USA, 24–31 March 2017; IEEE: Piscataway, NJ, USA, 2017.
24. Lepetit, V.; Moreno-Noguer, F.; Fua, P. Epnp: An accurate o (n) solution to the pnp problem. *Int. J. Comput. Vis.* **2009**, *81*, 155–166. [CrossRef]
25. Bleser, G.; Pastarmov, Y.; Stricker, D. Real-time 3d camera tracking for industrial augmented reality applications. In Proceedings of the 13th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision 2005 in Co-Operation with EUROGRAPHICS, Plzen, Czech Republic, January 31–February 4 2005.

26. Petit, A.; Marchand, E.; Kanani, K. Vision-based detection and tracking for space navigation in a rendezvous context. In *Int. Symp. on Artificial Intelligence, Robotics and Automation in Space*; i-SAIRAS: Turin, Italy, 2012.
27. Vacchetti, L.; Lepetit, V.; Fua, P. Combining edge and texture information for real-time accurate 3d camera tracking. In Proceedings of the Third IEEE and ACM International Symposium on Mixed and Augmented Reality, Arlington, VA, USA, 5 November 2004; IEEE: Piscataway, NJ, USA, 2004.
28. Lowe, D.G. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vis.* **2004**, *60*, 91–110. [CrossRef]
29. Wagner, D.; Reitmayr, G.; Mulloni, A.; Drummond, T.; Schmalstieg, D. Pose tracking from natural features on mobile phones. In Proceedings of the 2008 7th IEEE/ACM International Symposium on Mixed and Augmented Reality, Cambridge, UK, 15–18 September 2008; IEEE: Piscataway, NJ, USA, 2008.
30. Rad, M.; Lepetit, V. Bb8: A scalable, accurate, robust to partial occlusion method for predicting the 3d poses of challenging objects without using depth. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017.
31. Tekin, B.; Sinha, S.N.; Fua, P. Real-time seamless single shot 6d object pose prediction. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018.
32. Pavlakos, G.; Zhou, X.; Chan, A.; Derpanis, K.G.; Daniilidis, K. 6-dof object pose from semantic keypoints. In Proceedings of the 2017 IEEE International Conference on Robotics and Automation (ICRA), Singapore, 29 May–3 June 2017; IEEE: Piscataway, NJ, USA, 2017.
33. Oberweger, M.; Rad, M.; Lepetit, V. Making deep heatmaps robust to partial occlusions for 3d object pose estimation. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018.
34. Peng, S.; Liu, Y.; Huang, Q.; Zhou, X.; Bao, H. Pvnet: Pixel-wise voting network for 6dof pose estimation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019.
35. Pham, T.; Cai, M.; Reid, I. Real-time monocular object instance 6d pose estimation. In Proceedings of the 29th British Machine Vision Conference (BMVC 2018), Newcastle, UK, 3 September–6 September 2019.
36. Li, Y.; Wang, G.; Ji, X.; Xiang, Y.; Fox, D. Deepim: Deep iterative matching for 6d pose estimation. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018.
37. Kehl, W.; Manhardt, F.; Tombari, F.; Ilic, S.; Navab, N. Ssd-6d: Making rgb-based 3d detection and 6d pose estimation great again. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017.
38. Xiang, Y.; Schmidt, T.; Narayanan, V.; Fox, D. Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes. *arXiv* **2017**, arXiv:1711.00199.
39. Li, X.; Chen, G.; Blasch, E.; Pham, K. Detecting missile-like flying target from a distance in sequence images. In *Signal Processing, Sensor Fusion, and Target Recognition XVII*; SPIE: Paris, France, 2008; Volume 6968.
40. He, K.; Gkioxari, G.; Dollár, P.; Girshick, R. Mask r-cnn. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017.
41. PyTorch3D. Available online: https://pytorch3d.readthedocs.io/en/latest/overview.html# (accessed on 21 December 2023).
42. Grosgeorge, D.; Petitjean, C.; Dacher, J.-N.; Ruan, S. Graph cut segmentation with a statistical shape model in cardiac MRI. *Comput. Vis. Image Underst.* **2013**, *117*, 1027–1035. [CrossRef]
43. Cha, J.; Farhangi, M.M.; Dunlap, N.; Amini, A.A. Segmentation and tracking of lung nodules via graph-cuts incorporating shape prior and motion from 4D CT. *Med. Phys.* **2018**, *45*, 297–306. [CrossRef] [PubMed]
44. Jaderberg, M.; Simonyan, K.; Zisserman, A. Spatial transformer networks. In *Advances in Neural Information Processing Systems*; MIT Press: Cambridge, MA, USA, 2015; Volume 28.
45. Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv* **2014**, arXiv:1409.1556.
46. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2016.
47. Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going deeper with convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015.
48. Szegedy, C.; Vanhoucke, V.; Ioffe, S.; Shlens, J.; Wojna, Z. Rethinking the inception architecture for computer vision. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 7–30 June 2016.
49. Szegedy, C.; Ioffe, S.; Vanhoucke, V.; Alemi, A. Inception-v4, inception-resnet and the impact of residual connections on learning. In Proceedings of the Thirty-first AAAI Conference on Artificial Intelligence, Vancouver, BC, Canada, 20–27 February 2017.
50. Wu, Y.; Kirillov, A.; Massa Francisco Lo, W.-Y.; Girshick, R. Detectron2. 2019. Available online: https://github.com/facebookresearch/detectron2 (accessed on 21 December 2023).