Fig. 1. Current annotation practice versus annotation with GeoVisipedia. (a) An image of a facility annotated using standard practices. (b) A satellite image of the facility appears in the GeoVisipedia interface. A building is highlighted and the wiki page to the right appears, providing a wealth of information about the building. Image of facility ©2015 DigitalGlobe, NextView License, annotations provided by authors.

facility model onto satellite imagery of the facility, effectively segmenting the image into the objects represented in the 3D model. Annotations are linked to components of the 3D model and hence linked with objects in the image. PRISM automatically adjusts projections to account for satellite position, orientation and optical system, and provides very accurate projections. Using PRISM, information about objects in the facility becomes ubiquitous across an ensemble of facility imagery.

GeoVisipedia is similar to an Automated Image Annotation (AIA) algorithm but there are differences in approach and outcome. The objective of AIA algorithms is to automatically assign semantic labels to imagery to enable content-based information retrieval. It is a very active research area, for example see recent surveys such as [4], [5]. With regards to automated satellite image annotation, many efforts are concerned with labeling one or more semantic concepts in the image such as harbors, runways, mobile home park, shopping center, etc. [6], [7]. In contrast, GeoVisipedia enables knowledge dissemination from experts to non-experts. It automatically promulgates annotations to other imagery after manual initialization. GeoVisipedia provides dense, detailed annotations and allows users to communicate with experts through dialog in a wiki page.

The PRISM algorithm has many uses, but this paper focuses on how GeoVisipedia uses the algorithm to project annotations onto imagery. We begin by describing the image metadata used by PRISM and review Rational Polynomial Coefficients (RPCs, also known as Rapid Positioning Capability). We next describe how RPCs, and other image metadata such as the height and width of the image, are used to create a view matrix which transforms vertices of a 3D facility model into Normalized Device Coordinates (NDCs). The view matrix rotates and scales the 3D model into the same view of as that of the facility as seen through the optical system of the satellite. After model components have been transformed by the view matrix, they are projected onto a 2D image mask and overlaid onto the facility image. We conclude with examples illustrating the efficacy of the PRISM algorithm in projecting annotations, and discuss the efficiencies of using GeoVisipedia over standard practices and applications of PRISM to building training data for machine learning algorithms.

II. The PRISM Algorithm

The PRISM (PRojection Inferred by Satellite Metadata) algorithm constructs a view matrix that projects components of a 3D model onto an image to create a component mask. Component silhouettes in the mask are aligned with corresponding facility objects as viewed in the satellite image. This one-to-one correspondence between silhouettes of model components and facility objects enables GeoVisipedia to attach annotations to model components and have these annotations projected onto facility objects in the satellite image.

A visual depiction of the data flow in the PRISM algorithm is shown in Figure 2. A 3D model of a facility and a satellite image of the facility are illustrated on the left of the figure. Metadata provided with the satellite image is used to create a view matrix as illustrated in the center of the figure. The view matrix is used to transform the vertices of the 3D model into normalized device coordinates. A graphics pipeline projects the model onto a mask as depicted on the left of Figure 2 where model components are randomly colored to ease interpretation of the mask. As illustrated in the lower right portion of the figure, silhouettes of model components are closely aligned with corresponding facility objects when the mask is overlaid onto the image. The object mask effectively segments objects in the image that have a component in the facility model, and this segmentation is invariant to the angle

from which the satellite image was collected due to the design of the view matrix.

A 3D facility model is a critical input to the PRISM algorithm. It consists of a set of model components M_i which represent objects in the facility such as buildings, tanks, and cooling units. GeoVisipedia associates annotations, in the form of Wiki pages, with these components. Objects in the facility are modeled by combining polygons. Important features, such as corners and other interest points, are represented by a set of vertices V_i^k which are located at positions (X_i, Y_i, Z_i) in the model coordinate system. The origin of the model coordinate system is selected as a point readily identified in satellite imagery, such as the ground-level corner of a building. It is also necessary to know the geographic coordinates of this point, such as its latitude, longitude and elevation (ϕ_0, λ_0, h_0). A unit of measure, such as meters, is used to scale the model. A simple facility model, consisting of a building and effluent stack, is shown in Figure 3c. This figure illustrates the model coordinate system where the X-axis points to the East, the Y-axis points to the North, and the Z-axis points Up.

In the example facility described in this paper, the 3D facility model was constructed using a workflow designed to rapidly generate computer-aided design (CAD) models from multi-view imagery. Aerial photography of this facility was captured using a small unmanned aerial vehicle (UAV) equipped with a gimbal stabilized Sony RX100IV digital camera. The UAV was controlled with the open-source Pixhawk flight controller which enabled fully autonomous flights. The target overlap for images was greater than 60%. A high-fidelity 3D point cloud was rendered from the source photos using the Agisoft Photoscan photogrammetry software. A suite of custom algorithms was used in tandem with the commercial reverse engineering software Geomagic Design X to extract dimensionally-accurate CAD models of the individual facility structures and surrounding terrain. Tessellated mesh renderings of the CAD structures are used as the input model deck to the PRISM algorithm.

Two fully-autonomous flights were conducted. The objective of the first flight was to capture imagery spanning 360° around each building. The flight path took the aircraft around each of the primary buildings while the flight controller kept the nose of the aircraft pointed at the center of each building to ensure imagery was collected of each facet of the building. Images were captured every 10 meters. The objective of the second flight was to capture imagery of a 360° view of the facility. This flight path was circular with a radius of 0.5 km with the nose of the aircraft pointed at center of the facility complex. Images were capture every 20 meters. Data collection with two flights took approximately 30 minutes, followed by 8 hours for point cloud construction. (As an aside, software is now available that can complete the point cloud construction in approximately 30 minutes, albeit at lower resolution.) The automated software took approximately one hour to develop a rough model of the facility and another two days of effort was required to refine the model to its current state. The model contains approximately 50 objects such as buildings, storage tanks and effluent stacks. Projection of model components onto satellite imagery of the facility were found to overlay the objects very closely indicating a geospatially accurate model (see Figure 2).

Many of the imaging satellites operated by commercial vendors use push-broom imaging systems. This type of camera has a linear array of detectors and builds an image line-by-line through motion of the satellite. Vendors provide metadata with each image and this metadata includes a camera model of the satellite imaging system. The camera model is a set of equations that map ground coordinates (latitude, longitude, elevation) to image pixel coordinates (x, y). Camera models are very complex, and must consider a variety of terms including artifacts created by a push broom scanner, off-nadir image collection, curvature of the Earth, image distortions created by the satellite optical system, and the change in satellite altitude over the course of an image acquisition. These factors result in pixel-to-pixel variations in ground sample distance, as well as different sample distances in the column (x) and row (y) directions of the image.

Image metadata used by the PRISM algorithm to construct its view matrix includes Rational Polynomial Coefficients, scaling and offset parameters associated with RPCs, and image dimensions. The PRISM algorithm constructs a view matrix in two steps. First, a mapping between model coordinates (X_i, Y_i, Z_i) and image coordinates (x_i, y_i) is calculated. Second, scale factors are calculated and used to map image coordinates into normalized device coordinates. Before describing how the PRISM algorithm constructs its view matrix, we briefly digress to review RPCs.

RPCs have been used as camera models by a number of satellite imagery companies for many years. RPCs map geospatial coordinates, expressed as latitude, longitude and elevation (ϕ, λ, h), to image pixel coordinates (x, y) using cubic polynomials with quadratic cross-coupling terms. These cubic polynomials bundle all the image distortions into one set of equations, whose coefficients are unique to each image. The column x and row y (also referred to as sample and line) positions in a satellite image for a given latitude ϕ, longitude λ, and elevation h (above a specified ellipsoid such as WGS-84), are computed using RPCs and the following procedure. First, coordinates of a position on earth (ϕ, λ, h) are normalized to minimize the introduction of numerical errors during subsequent calculations and improve stability of calculations. These normalizations are:

$$\begin{aligned} P &= (\phi - \text{LATITUDE\_OFFSET})/\text{LATITUDE\_SCALE} \\ L &= (\lambda - \text{LONGITUDE\_OFFSET})/\text{LONGITUDE\_SCALE} \\ H &= (h - \text{HEIGHT\_OFFSET})/\text{HEIGHT\_SCALE} \end{aligned} \tag{1}$$

where the scaling and offset factors are provided in the image metadata. The normalized latitude, longitude and height are used to find normalized column and row coordinates (x_n, y_n)

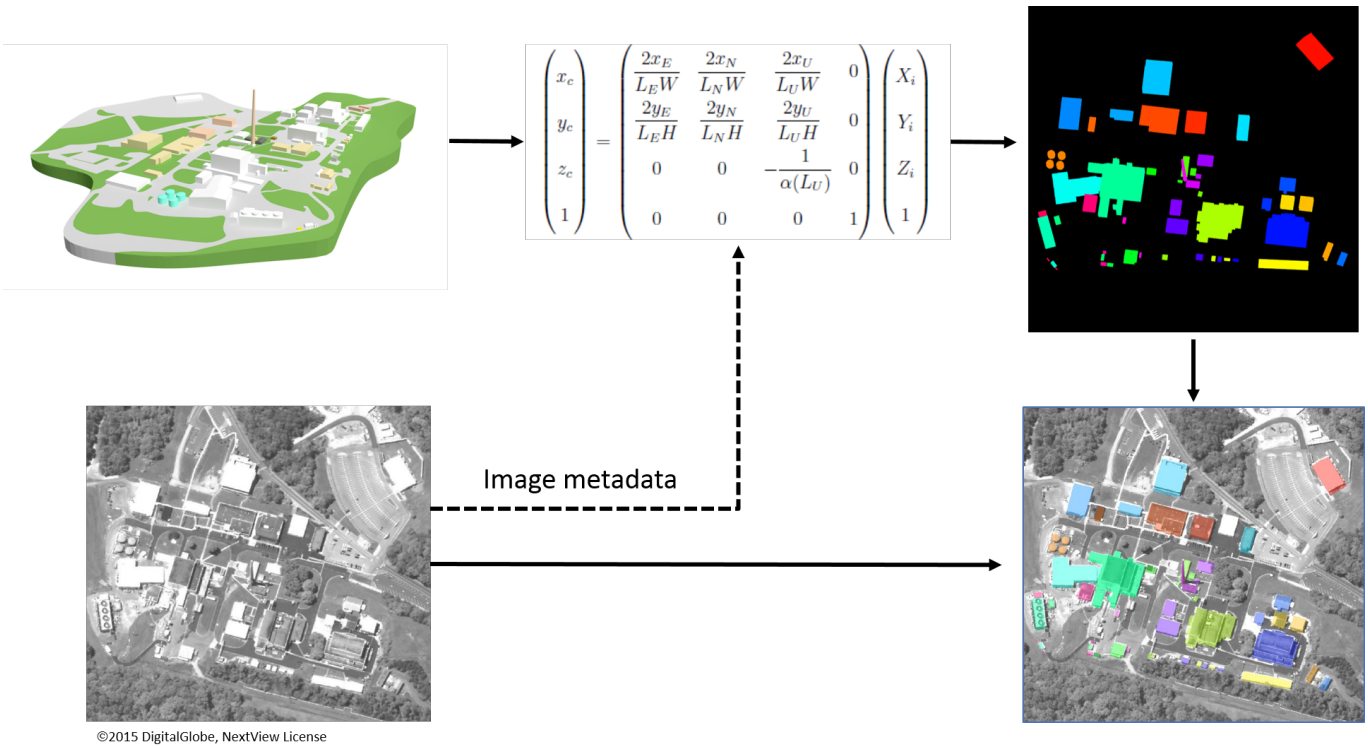

Fig. 2. Conceptual overview of the PRISM algorithm. A 3D facility model and satellite image of the facility are shown on the left. Metadata from the image is used to construct a view matrix that transforms components of the model into normalized device coordinates. Once the model has been transformed, components are projected onto a mask, with colors randomly assigned to objects. When the mask is overlaid on the image, model components are aligned with their associated facility objects as shown in the lower right corner. Facility image ©2015 DigitalGlobe, NextView License.

using rational polynomials and RPCs a_i, b_i, c_i, and d_i:

$$\begin{aligned} y_n &= \frac{\sum_{i=1}^{20} a_i \rho_i(P,L,H)}{\sum_{i=1}^{20} b_i \rho_i(P,L,H)} \\ x_n &= \frac{\sum_{i=1}^{20} c_i \rho_i(P,L,H)}{\sum_{i=1}^{20} d_i \rho_i(P,L,H)} \end{aligned} \tag{2}$$

We use the notation (x_n, y_n) to refer to a column and row position to emphasize that these positions are rational numbers and have fractional values. Rational function $\rho_i(\cdot)$ is cubic in (P, L, H) and expands, in the case of the numerator of y_n as:

$$\begin{aligned} &\sum_{i=1}^{20} a_i \rho_i(P,L,H) = \\ &a_1 + a_2 L + a_3 P + a_4 H + a_5 LP + a_6 LH + a_7 PH \\ &+ a_8 L^2 + a_9 P^2 + a_{10} H^2 + a_{11} PLH + a_{12} L^3 \\ &+ a_{13} LP^2 + a_{14} LH^2 + a_{15} L^2 P + a_{16} P^3 \\ &+ a_{17} PH^2 + a_{18} L^2 H + a_{19} P^2 H + a_{20} H^3 \end{aligned} \tag{3}$$

Similar expansions apply to the other summations in Equation 2. As noted, coordinates (x_n, y_n) are normalized and denormalization is necessary to find image coordinates (x, y) corresponding to earth location (ϕ, λ, h) in a satellite image:

$$\begin{aligned} y &= y_n \times \text{LINE\_SCALE} + \text{LINE\_OFFSET} \\ x &= x_n \times \text{SAMPLE\_SCALE} + \text{SAMPLE\_OFFSET} \end{aligned} \tag{4}$$

Further information on RPCs and how they are used as camera models in satellite imagery can be found in the papers by Tao [8] and Dial [9].

A satellite image is typically very large and difficult to manipulate, so a much smaller Area of Interest (AOI) is selected for processing. The AOI is centered at (x_0, y_0) in the satellite image and has dimensions of $W \times H$ pixels. Pixel boundaries of the AOI image are rounded to the nearest whole pixel since origin (x_0, y_0) is fractional. Coordinates in the satellite image (x_p, y_p) are related to coordinates in the AOI image (x_i, y_i) by:

$$\begin{aligned} x_i &= x_p - x_0 \\ y_i &= y_0 - y_p \end{aligned} \tag{5}$$

where the sign reversal accounts for the change of origin in the satellite image (upper left corner) to the origin in the AOI image (center of the image). See Figure 3a for a depiction of the AOI image within the satellite image. A 3D facility model is depicted in Figure 3b and has a coordinate system (X, Y, Z) centered at (ϕ, λ, h). In this coordinate system, the Y-axis is aligned to the North. The origin of the model is selected as a point with known latitude, longitude and height, and can be readily identified in satellite imagery. Coordinates of the origin of the AOI image (x_0, y_0) are located using RPC Equations 1—4 and translation Equations 5.

The PRISM algorithm uses RPC Equations 1—4 to construct a coordinate system at the origin of the AOI image (see

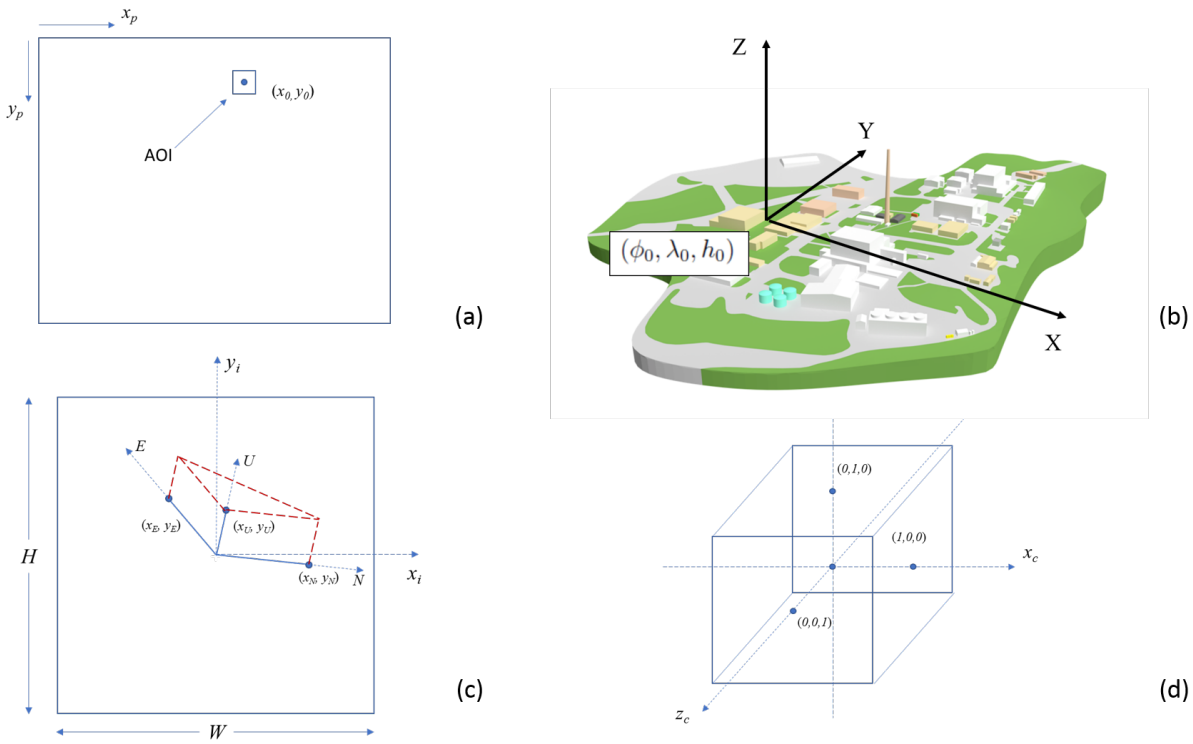

Fig. 3. Coordinate systems used in the prism algorithm. (a) Coordinate system of a satellite image with origin in upper left corner of image. An AOI image is defined centered at (x_0, y_0) The center of the AOI image corresponds to the center of the model coordinate system. (b) 3D model of a facility with coordinates (X, Y, Z) and origin of the coordinate system at (ϕ_0, λ_0, h_0) and the Y-axis associated with North. This point is typically selected as a easily identified point in the image, such as the base of a corner of a building and is associated with the center of the AOI image (x_0, y_0). (c) An AOI image of width W and height H. Using RPCs, a right prism is established with coordinates (E, N, U). (d) Clipping cube in normalized device coordinates.

Figure 3c). Coordinates (x_N, y_N), (x_E, y_E) and (x_U, y_U) are locations in the AOI image that are L_N, L_E and L_U meters from origin (x_0, y_0) to the North, East and Up, respectively. These points are located by finding the latitudes and longitudes of points that are L_N, L_E and L_U meters from origin (x_0, y_0) to the North, East and Up, respectively. Vincenty's formulae [10], or similar algorithm, is used to find the latitude and longitude of points L_E and L_N meters away from the origin in the x and y directions, and RPC Equations 1—4 are used to find the coordinates of these positions in the AOI image. The coordinates of (x_U, y_U) are found by adding L_U meters to z_0 and using the RPC equations to find the coordinates of this position in the AOI image. A value of 100 meters for lengths L_N, L_E and L_U has been found to work well in GeoVisipedia applications, but L_U needs to be scaled to prevent clipping tall objects. Note that points (x_N, y_N), (x_E, y_E), (x_U, y_U) and (x_0, y_0), when connected as depicted in Figure 3c, form a right prism and hence the name of the algorithm.

After a right-prism coordinate system has been established in the AOI image, coordinates (x_N, y_N), (x_E, y_E) and (x_U, y_U) are used to determine a projection matrix to map points in the 3D model into a 2D component mask. The PRISM algorithm uses an affine camera model to determine the entries of the projection matrix. An affine camera model is appropriate for this application since the depth of field (ground to the tallest object in the scene) is much smaller than the distance to the camera in the satellite. Consider the affine camera model $\mathbf{x} = \mathbf{P}_a\mathbf{X}$ which maps 3D homogeneous coordinates in the model $\mathbf{X}_i^T = (X_i, Y_i, Z_i, 1)$ into 2D homogeneous coordinates in an image $\mathbf{x}_i^T = (x_i, y_i, 1)$

$$\begin{pmatrix} x_i \\ y_i \\ 1 \end{pmatrix} = \begin{pmatrix} p_{11} & p_{12} & p_{13} & 0 \\ p_{21} & p_{22} & p_{23} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} X_i \\ Y_i \\ Z_i \\ 1 \end{pmatrix} \tag{6}$$

The PRISM algorithm sets entries p_{14} and p_{24}, which are used to describe translations, to zero since the origin of the AOI image has already been translated to the center of the AOI image (see Equations 5). Expanding Equation 6 we have

$$\begin{aligned} x_i &= p_{11}X_i + p_{12}Y_i + p_{13}Z_i \\ y_i &= p_{21}X_i + p_{22}Y_i + p_{23}Z_i \end{aligned} \tag{7}$$

Using the prism coordinates we have the following correspondences between the model and the image

$$\begin{aligned} (L_E, 0, 0) &\rightarrow (x_E, y_E) \\ (0, L_N, 0) &\rightarrow (x_N, y_N) \\ (0, 0, L_U) &\rightarrow (x_U, y_U) \end{aligned}$$

where the x, y, and z directions are associated with East, North and Up respectively. Substituting these correspondences into Equations 7 yields

$$\begin{aligned} x_E &= p_{11} L_E \\ y_E &= p_{21} L_E \\ x_N &= p_{12} L_N \\ y_N &= p_{22} L_N \\ x_U &= p_{13} L_U \\ y_U &= p_{23} L_U \end{aligned} \tag{8}$$

Inserting these parameters into affine projection matrix $\mathbf{P}_a$ (with p_{14} and p_{24} set to zero) gives

$$\mathbf{P}_a = \begin{pmatrix} \frac{x_E}{L_E} & \frac{x_N}{L_N} & \frac{x_U}{L_U} & 0 \\ \frac{y_E}{L_E} & \frac{y_N}{L_N} & \frac{y_U}{L_U} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{9}$$

At this point we are able to project vertices in the model $\mathbf{X}_i^T = (X_i, Y_i, Z_i, 1)$ into points in the AOI image $\mathbf{x}_i^T = (x_i, y_i, 1)$ using projection matrix $\mathbf{P}_a$ in Equation 9. The next step is to devise scaling factors that will multiply $\mathbf{P}_a$ to transform points in the model into normalized device coordinates.

To render model vertices into polygons and project the polygons onto the AOI image, it is necessary to scale model coordinates into normalized device coordinates (x_c, y_c, z_c). Normalized device coordinates are used by graphics pipelines and scale projected model coordinates into a $2 \times 2 \times 2$ cube centered at the origin. (See Figure 3d for a depiction of normalized device coordinates and clipping cube.) Since graphics pipelines use homogeneous coordinate systems $(x_c, y_c, z_c, 1)$, projection matrix $\mathbf{P}_a$ is modified by inserting an additional row $(0, 0, 1, 0)$ between the second and third rows to account for the z_c coordinate:

$$\mathbf{P} = \left(\begin{array}{cccc} \frac{x_E}{L_E} & \frac{x_N}{L_N} & \frac{x_U}{L_U} & 0 \\ \frac{y_E}{L_E} & \frac{y_N}{L_N} & \frac{y_U}{L_U} & 0 \\ \hdashline 0 & 0 & 1 & 0 \\ \hdashline 0 & 0 & 0 & 1 \end{array}\right) \tag{10}$$

Scaling model coordinates to normalized device coordinates is accomplished with scaling factors

$$s_x = \frac{2}{W} \tag{11}$$

$$s_y = \frac{2}{H} \tag{12}$$

$$s_z = \frac{-1}{\alpha L_U} \tag{13}$$

The factor of two in Equations 11 and 12 accounts for the dimensions of the NDC clipping cube. If objects in the z-direction are greater than m meters in height above the ellipsoid (we set $m = 100$ meters), they will be clipped. Scaling factor s_z is designed to scale the 3D model into the NDC clipping cube and a reasonable scaling factor is $s_z = -2/L_U$ where the negative sign is due to a reversal in the z_c coordinate direction. But, it was determined in practice that additional control over the scaling was needed to prevent tall objects from being clipped by the graphics pipeline. Hence, a scaling factor of α was used and for satisfactory results is set such that $\alpha > 1.0$. Applying scaling factors to the projection matrix $\mathbf{P}_a$ yields our final matrix:

$$\mathbf{P}_c = \begin{pmatrix} \frac{2}{W} & 0 & 0 & 0 \\ 0 & \frac{2}{H} & 0 & 0 \\ 0 & 0 & \frac{-1}{\alpha L_U} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \frac{x_E}{L_E} & \frac{x_N}{L_N} & \frac{x_U}{L_U} & 0 \\ \frac{y_E}{L_E} & \frac{y_N}{L_N} & \frac{y_U}{L_U} & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{14}$$

The matrix multiplication yields the final projection equation

$$\mathbf{P}_c = \begin{pmatrix} \frac{2x_E}{L_E W} & \frac{2x_N}{L_N W} & \frac{2x_U}{L_U W} & 0 \\ \frac{2y_E}{L_E H} & \frac{2y_N}{L_N H} & \frac{2y_U}{L_U H} & 0 \\ 0 & 0 & -\frac{1}{\alpha(L_U)} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{15}$$

Thus, transformation from homogeneous model coordinates to homogeneous device coordinates is given by

$$\begin{pmatrix} x_c \\ y_c \\ z_c \\ 1 \end{pmatrix} = \begin{pmatrix} \frac{2x_E}{L_E W} & \frac{2x_N}{L_N W} & \frac{2x_U}{L_U W} & 0 \\ \frac{2y_E}{L_E H} & \frac{2y_N}{L_N H} & \frac{2y_U}{L_U H} & 0 \\ 0 & 0 & -\frac{1}{\alpha(L_U)} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} X_i \\ Y_i \\ Z_i \\ 1 \end{pmatrix} \tag{16}$$

Equation 16 is the PRISM view matrix and performs the projection and scaling operations that transform homogeneous model coordinates into normalized device coordinates. These coordinates are used by a graphics pipeline to assemble vertices V_i^k from model object M_i into polygons. The component mask shown in the upper right Figure 2 is produced by ignoring the z_c coordinate: $(x_i, y_i, z_i, 1) \rightarrow (x_i, y_i)$. Colors are randomly assigned to the polygons in the component mask to ease interpretation. The alpha channel of the component mask is adjusted to allow minute details of the image to show through the mask enabling the user to see fine details on facility objects. When the component mask and satellite image are sent to a rendering engine, the combined image appears as the lower right of Figure 2.

Fig. 4. Illustration of the performance of the PRISM algorithm to project model objects onto a satellite image. The region of interest in the left image contains an effluent stack, and the images on the right show how the prism algorithm is able to project a model of the stack onto the four different satellite images. Facility image ©2015 DigitalGlobe, NextView License.

Figure 4 illustrates the efficacy of the PRISM algorithm in projection model components that have severe parallax due to the image collection geometry. The image on the left on Figure 4 is a satellite image of a facility, and the region of interest (indicated by a yellow box) contains an effluent stack. The four images on the right illustrate the projection of the stack model onto four different satellite images of the effluent stack. The four image collection geometries of the satellite result in four different orientations of the stack in the imagery. As seen in the figure, the PRISM algorithm is able to correctly project the stack model onto the imagery despite the parallax.

With regards to the computational efficiency of the PRISM algorithm it is important to note that RPCs are relatively accurate but are completely incompatible with graphics rendering pipelines. Our goal is to use the RPC equations to generate a locally accurate 4×4 view matrix that is compatible with standard rendering pipelines, thereby enabling us to make use of standard rendering tools. This simplification is possible due to the insight that, because of the imaging platforms large standoff distance, local areas in the image (out to several hundred meters) can be treated as parallel projections. PRISM pre-calculates a single 4×4 view matrix using very straightforward calculations as derived above. Instead of attempting to slowly render or resample images using RPCs, PRISM can render tens of frames a second of projected models using its view matrix and can perform these projections to sub-pixel accuracy.

III. Discussion

The efficiency of using GeoVisipedia to annotate imagery and the application of the PRISM algorithm to machine learning are closely related. GeoVisipedia links information to components of a 3D geospatial model and uses PRISM to project masks of the components from the 3D model onto imagery. Information linked to the 3D model is thus linked to the object in the image. The initial cost of using GeoVisipedia and the PRISM algorithm is creating a 3D scene model with sufficient geospatial precision and accuracy to adequately resolve objects of interest. Once this cost is accepted it can be amortized across potentially very large amounts of imagery. Given the initialization costs of using GeoVisipedia, it is not well suited for one-time annotations of facilities with a very limited set of satellite imagery. GeoVisipedia is an appropriate technology if the goal is to collect, discuss and integrate comments to develop a comprehensive understanding of a facility.

The PRISM algorithm has an obvious application of generating labeled data for training machine learning algorithms. Consider a fixed-position object of interest, such as a particular type of equipment or storage tank. If a 3D facility model contains the component associated with the object, then the PRISM algorithm is used to create a mask of the object for each image in an ensemble of facility imagery. Mask creation is very quick, processing tens of images per second as noted in the previous section. A fixed-sized rectangular region is centered on the object mask to create a properly dimensioned sub image. These sub images form the training set for that object. Given a diverse collection of satellite imagery, the training set contains multiple viewing angles, shadows, glint and clutter needed to build a robust training set.

IV. Acknowledgments

Figures 3 and 4 are original artwork by the authors and reprinted from [2] with permission from the publisher. This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344. LLNL-CONF-XXXXXX.