Geometric and Physical Constraints for Drone-Based Head Plane Crowd Density Estimation

Weizhe Liu, Krzysztof Lis, Mathieu Salzmann, Pascal Fua

***Abstract*— State-of-the-art methods for counting people in crowded scenes rely on deep networks to estimate crowd density in the image plane. While useful for this purpose, this image-plane density has no immediate physical meaning because it is subject to perspective distortion. This is a concern in sequences acquired by drones because the viewpoint changes often. This distortion is usually handled implicitly by either learning scale-invariant features or estimating density in patches of different sizes, neither of which accounts for the fact that scale changes must be consistent over the whole scene.**

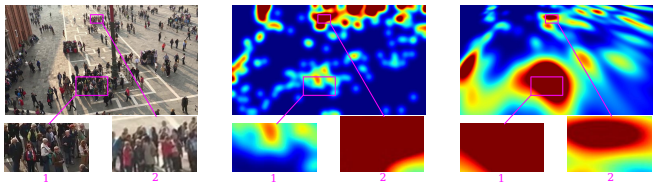
In this paper, we explicitly model the scale changes and reason in terms of people per square-meter. We show that feeding the perspective model to the network allows us to enforce global scale consistency and that this model can be obtained on the fly from the drone sensors. In addition, it also enables us to enforce physically-inspired temporal consistency constraints that do not have to be learned. This yields an algorithm that outperforms state-of-the-art methods in inferring crowd density from a moving drone camera especially when perspective effects are strong.

I. Introduction

With the growing prevalence of drones, drone-based crowd density estimation becomes increasingly relevant to applications such as autonomous landing and video surveillance. In recent years, the emphasis has been on developing *counting-by-density* algorithms that rely on regressors trained to estimate the density of crowd per unit area so that the total numbers of people can be obtained by integration, without explicit detection being required. The regressors can be based on Random Forests [1], Gaussian Processes [2], or more recently Deep Nets [3], [4], [5], [6], [7], [8], [9], [10], [11], [12], [13], [14], [15], [16], [17], with most state-of-the-art approaches now relying on the latter.

While effective, these algorithms all estimate density in the image plane. As a consequence, and as can be seen in Fig. 1(a,b), two regions of the scene containing the same number of people per square meter can be assigned different densities. However, for the purposes of autonomous landing or crowd size estimation, the density of people on the ground is a more relevant measure and is *not* subject to such distortions, as shown in Fig. 1(c).

In this paper, we therefore introduce a crowd density estimation method that *explicitly* accounts for perspective distortion to produce a real-world density map, as opposed to an image-based one. To this end, it takes advantage of the fact that drone cameras can be naturally registered to the scene using the drone's internal sensors, which as we will see are accurate enough for our purposes. This contrasts with methods that *implicitly* deal with perspective effects by either learning scale-invariant features [4], [6], [8] or estimating density in patches of different sizes [5]. Unlike these, we model perspective distortion globally and account for the fact that people's projected size changes consistently across the image. To this end, we feed to our density-estimation CNN not only the original image but also an identically-sized image that contains the local scale, which is a function of the camera orientation with respect to the ground plane.



(a) (b) (c)

Fig. 1: **Measuring people density.** **(a)** An image of Piazza San Marco in Venice. The two purple boxes highlight patches in which the crowd density per square meter is similar. **(b)** Ground-truth *image density* obtained by averaging the head annotations in the image plane. The two patches are in the same locations as in (a). The density per square pixel strongly differs due to perspective distortion: the farther patch 2 wrongly features a higher density than closer patch 1, even though the people do not stand any closer to each other. **(c)** By contrast the ground-truth *head plane density* introduced in Section III-A is unaffected by perspective distortion. The density in the two patches now has similar peak values, as it should.

An additional benefit of reasoning in the real world is that we can encode physical constraints to model the motion of people in a video sequence. Specifically, given a short sequence as input to our network, we impose temporal consistency by forcing the densities in the various images to correspond to physically possible people flows. In other words, we *explicitly* model the motion of people, with physically-justified constraints, instead of *implicitly* learning long-term dependencies only across annotated frames, which are typically sparse over time, via LSTMs, as is commonly done in the literature [7].

Our contribution is therefore an approach that incorporates geometric and physical constraints directly into an end-to-end learning formalism for crowd counting using information directly obtained from the drone sensors. As evidenced by

The authors are with the Computer Vision Laboratory, School of Communication and Computer Sciences, École Polytechnique Fédérale de Lausanne (EPFL), Switzerland.

This work was supported in part by a grant from the Swiss Federal Office for Defense Procurement.

our experiments, this enables us to outperform the state-of-the-art on a drone-based video sequences with severe perspective distortion.

II. Related work

Early crowd counting methods [18], [19] tended to rely on *counting-by-detection*, that is, explicitly detecting individual heads or bodies and then counting them. Unfortunately, in very crowded scenes, occlusions make detection difficult, and these approaches have been largely displaced by *counting-by-density-estimation* ones, which rely on training a regressor to estimate people density in various parts of the image and then integrating. This trend started with [1], and [2], which used Random Forests and Gaussian Process regressors, respectively.

Even though approaches such as these early ones that rely on low-level features—a survey of which can be found in [8]—can be effective they have now mostly been superseded by CNN-based methods. The same can be said about methods that count objects instead of people [20], [21].

a) **Perspective Distortion:** Earlier approaches to handling such distortions [3] involve regressing to both a crowd count and a density map. Unlike ours that passes a perspective map as an input to the deep network, they use the perspective map to compute a metric and use it to retrieve candidate training scenes with similar distortions before tuning the model. This complicates training, which is not end-to-end, and decreases performance.

These approaches were recently extended by [6], whose *SwitchCNN* exploits a classifier that greedily chooses the sub-network that yields the best crowd counting performance. Max pooling is used extensively to down-scale the density map output, which improves the overall accuracy of the counts but decreases that of the density maps as pooling incurs a loss in localization precision.

Perspective distortion is also addressed in [5] via a scale-aware model called *HydraCNN*, which uses different-sized patches as input to the CNN to achieve scale-invariance. To the same end, different kernel sizes are used in [4] and in [15] features from different layers are extracted instead. In the recent method of [8], a network dubbed CP-CNN combines local and global information obtained by learning density at different resolutions. It also accounts for density map quality by adding extra information about the pre-defined density level of different patches and images. While useful, this information is highly scene specific and would make generalization difficult. More recent works use different techniques, such as a growing CNN [12], fusing crowd counting with people detection [10], adding a new measurement between prediction and ground truth density map [17], using a scale-consistency regularizer [9], employing a pool of decorrelated regressors [13], refining the density map in an iterative process [16], leveraging web-based unlabeled data [14], to further boost performance. However, none of them is specifically designed to handle perspective effects.

In any event, all the approaches mentioned above rely on the network learning about perspective effects without explicitly modeling them. As evidenced by our results, this is suboptimal given the finite amounts of data available in practical situations. Furthermore, while learning about perspective effects to account for the varying people sizes, these methods still predict density in the image plane, thus leading to the unnatural phenomenon that real-world regions with the same number of people are assigned different densities, as shown in Fig. 1(b). By contrast, we produce densities expressed in terms of number of people per square meter of ground, such as the ones shown in Fig. 1(c), and thus are immune to this problem.

b) **Temporal Consistency:** The recent method of [7] is representative of current approaches in enforcing temporal consistency by incorporating an LSTM module [22] to perform feature fusion over time. This helps but can only capture temporal correlation across annotated frames, which are widely separated in most existing training sets. In other words, the network can only learn long-term dependencies at the expense of shorter-term ones.

By contrast, since we reason about crowd density in real world space, we can model physically-correct temporal dependencies via frame-to-frame feasibility constraints, without any additional annotations.

III. Perspective Distortion

All existing approaches estimate the crowd density in the image plane and in terms of people per square pixel, which changes across the image even if the true crowd density per square meter is constant. For example, in many scenes such as the one of Fig. 1(a), the people density in farther regions is higher than that in closer regions, as can be seen in Fig. 1(b).

In this work, we train the system to directly predict the crowd density in the physical world, which does not suffer from this problem and is therefore unaffected by perspective distortion, assuming that people are standing on an approximately flat surface. Our approach could easily be extended to a non flat one given a terrain model. In a crowded scene, people's heads are more often visible than their feet. Consequently, it is a common practice to provide annotations in the form of a dot on the head for supervision purposes. To account for this, we define a *head plane*, parallel to the ground and lifted above it by the average person's height. We assume that the camera has been calibrated so that we are given the homography between the image and the head plane.

A. Image Plane versus Head Plane Density

Let $\mathbf{H}_i$ be the homography from an image I_i to its corresponding head plane. We define the ground-truth density as a sum of Gaussian kernels centered on peoples' heads in the head plane. Because we work in the physical world, we can use the same kernel size across the entire scene and across all scenes. A head annotation P_i, that is, a 2D image point expressed in projective coordinates, is mapped to $\mathbf{H}_i P_i$ in the head plane. Given a set $A_i = \{P_i^1, ..., P_i^{c_i}\}$ of c_i such annotations, we take the *head plane density* G_i' at point P'

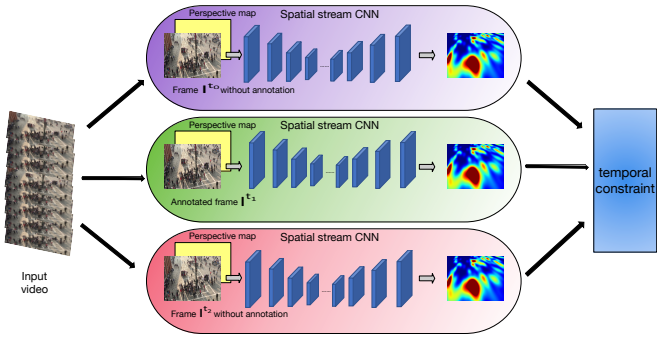


Fig. 2: **Three-stream architecture.** A spatial stream is a CSRNet [11] with 3 transposed convolutional layers, that takes as input the image and a perspective map. It is duplicated three times to process images taken at different times and minimize a loss that enforces temporal consistency constraints.

expressed in head plane coordinates to be

$$G'_i(P') = \sum_{j=1}^{c_i} \mathcal{N}(P'; \mathbf{H}_i P_i^j, \sigma) \ , \tag{1}$$

where $\mathcal{N}(.; \mu, \sigma)$ is a 2D Gaussian kernel with mean μ and variance σ. We can then map this head plane density to the image coordinates, which yields a density at pixel location P given by

$$G_i(P) = G'_i(\mathbf{H}_i P) \ . \tag{2}$$

An example density G_i is shown in Fig. 1(c). Note that, while the density is Gaussian in the head plane, it is *not* in the image plane.

B. Geometry-Aware Crowd Counting

Since the head plane density map can be transformed into an image of the same size as that of the original image, we could simply train a deep network to take a 3-channel RGB image as input and output the corresponding density map. However, this would mean neglecting the geometry encoded by the ground plane homography, namely the fact that the local scale does not vary arbitrarily across the image and must remain globally consistent.

To account for this, we associate to each image I a *perspective map* M of the same size as I containing the local scale of each pixel, that is, the factor by which a small area around the pixel is multiplied when projected to the head plane. We then use a convolutional network with 4 input channels instead of only 3. The first three are the usual RGB channels, while the fourth contains the perspective map. We will show in the result section that this substantially increases accuracy over using the RGB channels only. This network is one of the *spatial streams* depicted by Fig. 2. To learn its weights Θ, we minimize the *head plane loss* $L_H(I, M, G; \Theta)$, which we take to be the mean square error between the predicted head plane density and the ground-truth one.

To compute the perspective map M, let us first consider the image pixel $(x, y)^\intercal$ and an infinitesimal area $dx\ dy$ surrounding it. Let $(x', y')^\intercal$ and $dx'dy'$ be their respective projections on the head plane. We take $M(x, y)$, the scale at $(x, y)^\intercal$, to be $(dx'dy')/(dx\ dy)$, which we compute as follows. Using the variable substitution equation, we write

$$dx'dy' = |\det(J(x, y))| dx\ dy \ , \tag{3}$$

where $J(x, y)$ is the Jacobian matrix of the coordinate transformation at the point $(x, y)^\intercal$:

$$J = \begin{bmatrix} \frac{\partial x'}{\partial x} & \frac{\partial x'}{\partial y} \\ \frac{\partial y'}{\partial x} & \frac{\partial y'}{\partial y} \end{bmatrix} . \tag{4}$$

The scale map M is therefore equal to

$$M(x, y) = |\det(J(x, y))|. \tag{5}$$

The detailed solution can be found in [23]. Eq. 5 enables us to compute the perspective map that we use as an input to our network, as discussed above. It also allows us to convert between people density F in image space, that is, people per square pixel, and people density G' on the head plane. More precisely, let us consider a surface element dS in the image around point $(x, y)^\intercal$. It is scaled by $\mathbf{H}$ into $dS' = M(x, y)dS$. Since the projection does not change the number of people, we have

$$\begin{aligned} F(x, y)dS &= G'(x', y')dS' = G'(x', y')M(x, y)dS \\ \Rightarrow F(x, y) &= M(x, y)G'(x', y') \ . \end{aligned} \tag{6}$$

Expressed in image coordinates, this becomes

$$F(x, y) = M(x, y)G(x, y) \ , \tag{7}$$

which we use in the results section to compare our algorithm that produces head plane densities against the baselines that estimate image plane densities.

C. Obtaining scene geometry from UAV sensors

We calculate the homography matrix $\mathbf{H}$ using the camera's altitude h and pitch angle θ reported by the UAV sensors. We choose the world coordinate frame such that the head plane is given by $Z = 0$ and the origin $(0, 0, 0)^\intercal$ is directly under the UAV. The camera extrinsics are described by the rotation matrix $R = R_y(\frac{\pi}{2} + \theta)$ and translation vector $t = (0, 0, h)^\intercal$.

The relation between a point $(x_h, y_h, 0)^\intercal$ on the head plane and its projection $(u, v)^\intercal$ onto the image is expressed by the following equation, in homogenous coordinates:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = K \begin{bmatrix} R_{11} & R_{12} & R_{13} & t_1 \\ R_{21} & R_{22} & R_{23} & t_2 \\ R_{31} & R_{32} & R_{33} & t_3 \end{bmatrix} \begin{bmatrix} x_h \\ y_h \\ 0 \\ 1 \end{bmatrix} , \tag{8}$$

where K is the camera's intrinsic matrix and $w \neq 0$ is an arbitrary scale factor. Solving for $(x_h, y_h)^\intercal$ we obtain:

$$\begin{bmatrix} x_p \\ y_p \\ 1 \end{bmatrix} = w \left(K \begin{bmatrix} R_{11} & R_{12} & t_1 \\ R_{21} & R_{22} & t_2 \\ R_{31} & R_{32} & t_3 \end{bmatrix} \right)^{-1} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} . \tag{9}$$

The transformation from the image to the head plane is therefore given by the homography $\mathbf{H} = \left(K \left[\ R_1 \ | \ R_2 \ | \ t \ \right]\right)^{-1}$.

IV. Temporal Consistency

The spatial stream network depicted at the top of Fig. 2 operates on single frames of a video sequence. To increase robustness, we now show how to enforce temporal consistency across triplets of frames. Unlike in an LSTM-based approach, such as [7], we can do this across any three frames instead of only across annotated frames. Furthermore, by working in the real world plane instead of the image plane, we can explicitly exploit physical constraints on people's motion.

A. People Conservation

An important constraint is that people do not appear or disappear from the head plane except at the edges or at specific points that can be marked as exits or entrances. To model this, we partition the head plane into K blocks. Let $N(k)$ for $1 \leq k \leq K$ denote the neighborhood of block B_k, including B_k itself. Let m_k^t be the number of people in B_k at time t and let $t_0 < t_1 < t_2$ be three different time instants. In experiments, we empirically set the block size to 30 by 30 pixels.

If we take the blocks to be large enough for people not be able to traverse more than one block between two time instants, people in the interior blocks can only come from a block in $N(k)$ at the previous instant and move to a block in $N(k)$ at the next. As a consequence, we can write

$$\forall k \quad m_k^{t_1} \leq \sum_{i \in N(k)} m_i^{t_0} \text{ and } m_k^{t_1} \leq \sum_{i \in N(k)} m_i^{t_2} \; . \qquad (10)$$

In fact, an even stronger equality constraint could be imposed as in [24] by explicitly modeling people flows from one block to the next with additional variables predicted by the network. However, not only would this increase the number of variables to be estimated, but it would also require enforcing hard constraints between different network's outputs.

In practice, since our networks output head plane densities, we write

$$m_k^t = \sum_{(x',y')^\intercal \in B_k} \hat{G}'^t(x', y') \; , \qquad (11)$$

where $\hat{G}'^t$ is the predicted people density at time t, as defined in Section III-B. This allows us to reformulate the constraints of Eq. 10 in terms of densities.

B. Siamese architecture

To enforce these constraints, we introduce the siamese architecture depicted by Fig. 2, with weights Θ. It comprises three identical streams, each stream is a CSRNet [11] with 3 transposed convolutional layers added before the last convolutional layer, so that the input image and output density map have the same size. These three identical steams take as input images acquired at times t_0, t_1, and t_2 along with their corresponding perspective maps, as described in Section III-B. Each one produces a head plane density estimate G'^{t_i} and we define the temporal loss term $L_T(I^{t_0}, I^{t_1}, I^{t_2}, M^{t_0}, M^{t_1}, M^{t_2}; \Theta)$ as

$$\frac{1}{2K} \sum_{k=1}^{K} [(max(0, m_k^{t_1} - U_k^{t_0}))^2 + (max(0, m_k^{t_1} - U_k^{t_2}))^2] \; , \qquad (12)$$

where m_k^t is the sum of predicted densities in block B_k, as in Eq. 11, and $U_k^t = \sum_{i \in N(k)} m_i^t$ is the sum of densities in the neighborhood of B_k.

In other words, L_T penalizes violations of the constraints of Eq. 10. At training time, we minimize the composite loss

$$L_H(I^{t_1}, M^{t_1}, G'^{t_1}; \Theta) + L_T(I^{t_0}, I^{t_1}, I^{t_2}, M^{t_0}, M^{t_1}, M^{t_2}; \Theta), \qquad (13)$$

where L_H is the head plane loss introduced in Section III-B. Since the loss requires the ground truth density only for frame I^{t_1}, we only need annotations for that frame. Therefore, we can use arbitrarily-spaced and unannotated frames to impose temporal consistency and improve robustness, which is not something LSTM-based methods can do.

V. Experiments

A. Datasets and Experimental Setup

Our approach is designed to handle perspective effects as well as to enforce temporal consistency. As there is no publicly available drone-based crowd counting dataset, we filmed a six-minute long sequence using a DJI phantom 4 pro drone flying over a university campus and filming it from many different perspectives. We manually annotated 90 images such as the one of Fig. 3 and used 54 of them for training and validation purposes and the remainder for testing. The people count ranges from 54 to 301 in this dataset. We will refer to it as **Campus**. In the supplementary material, we provide a video showing our results on a subsequence.

To demonstrate that our approach also works in a very different context, we also evaluate it on the publicly available **Venice** [25] dataset, which was recorded using a mobile phone. It features Piazza San Marco as seen from various viewpoints on the second floor of the basilica and substantial perspective effects. This dataset comprises 4 different sequences and 167 annotated frames. Fig. 1 depicts one of these. The white lines on the Piazza make it easy to estimate the plane homography using standard photogrammetric techniques and the sequence is thus a good proxy for drone-acquired footage.

We focus on head-plane and ground-plane densities, as opposed to image-plane densities, because they are the ones that have a true physical meaning independently of the camera motion. In this section, we therefore report our results and baselines ones in head-plane density terms. However, we also provides image plane density results to demonstrate that our model outperforms the baselines in both cases.

B. Baselines

We benchmark our approach against three recent methods for which the code is publicly available: **CSRNet** [11], **MCNN** [4] and **SwitchCNN** [6]. As discussed in the related work section, they are representative of current approaches to handling the fact that people's sizes vary depending on their distance to the camera.

We will refer to our complete approach as **OURS**. To tease out the individual contributions of its components, we

also evaluate two degraded versions of it. **OURS-NoGeom** uses the CNN to predict densities but does not feed it the perspective map as input. **OURS-GeomOnly** uses the full approach described in Section III but does not impose temporal consistency.

C. Evaluation Metrics

Most previous works in crowd density estimation use mean absolute error (MAE) and root mean squared error (RMSE) as their evaluation metric. They are defined as

$$\text{MAE} = \frac{1}{N}\sum_{1}^{N}|z_i - \hat{z}_i| \text{ and RMSE} = \sqrt{\frac{1}{N}\sum_{1}^{N}(z_i - \hat{z}_i)^2}, \quad (14)$$

where N is the number of test images, z_i denotes the true number of people inside the ROI of the ith image and $\hat{z}_i$ the estimated number of people. While indicative, these two metrics are very coarse, since these two metrics only take into consideration the total number of people irrespective of where in the scene they may be, so they are incapable of evaluating the correctness of the spatial distribution of crowd density. A false positive in one region, coupled with a false negative in another, can still yield a perfect total number of people.

We therefore introduce one additional metric that provide finer grained measures, accounting for localization errors. We name it the mean pixel-level absolute error (MPAE) and take it to be

$$\text{MPAE} = \frac{\sum_{i=1}^{N}\sum_{j=1}^{H}\sum_{k=1}^{W}|D_{i,j,k} - \hat{D}_{i,j,k}| \times \mathbf{1}_{\{D_{i,j,k} \in R_i\}}}{N}, \quad (15)$$

where $D_{i,j,k}$ is the ground-truth density of the ith image at pixel (j,k), $\hat{D}_{i,j,k}$ is the corresponding estimated density, R_i is the ROI of the ith image, $\mathbf{1}_{\{.\}}$ is the indicator function, and W and H are the image dimensions. MPAE quantifies how wrongly localized the densities are.

The baseline models [4], [11], [6] are designed to predict density in the image plane instead of the head plane, as our model does. Fortunately, the densities in image plane and head plane can be easily converted into each other, as shown in Section III. For a fair comparison, we therefore train the baseline models [4], [11], [6] as reported in original the papers to estimate density in the image-plane. We then used Eq. 7 to convert to head-plane density. Thus we can use the MAE, RMSE, and MPAE metrics to compare both kinds of densities.

D. Quantitative Evaluation

We report our comparative results in Tables I, II, III and IV. Enforcing temporal consistency requires the central frame to be annotated but the other two can be chosen arbitrarily. When running **OURS**, that is, enforcing both geometry and temporal constraints, we used triplets of images temporally separated by 1, 5, or 10 frames. We provide a qualitative comparison in Fig. 3.

In Tables I and III, we used Eq. 7 to convert the image plane densities computed by the baselines into head-plane densities that can be compared to ours. In Tables II and IV, we instead converted our head plane densities into image plane ones that can be compared to theirs. Either way, **OURS-GeomOnly** outperforms the baselines. Furthermore, imposing temporal consistency gives our approach a further boost.

Model	MAE	$RMSE$	$MPAE$
CSRNet [11]	50.1	54.2	125.6
MCNN [4]	23.5	30.6	143.9
SwitchCNN [6]	91.0	120.5	330.1
OURS-NoGeom	29.2	34.8	131.2
OURS-GeomOnly	20.1	24.7	135.1
OURS (frame interval 1)	**11.9**	**15.1**	116.9
OURS (frame interval 5)	16.1	20.2	**113.2**
OURS (frame interval 10)	13.4	17.2	126.2

TABLE I: Comparative results in terms of head plane crowd density on the **Campus** dataset.

Model	MAE	$RMSE$	$MPAE$
CSRNet [11]	51.3	57.6	126.4
MCNN [4]	24.2	37.1	146.2
SwitchCNN [6]	91.7	122.1	340.7
OURS-NoGeom	29.8	35.2	132.0
OURS-GeomOnly	21.2	24.7	136.8
OURS (frame interval 1)	**12.3**	**16.0**	117.3
OURS (frame interval 5)	16.9	22.3	**114.1**
OURS (frame interval 10)	14.2	18.0	128.7

TABLE II: Comparative results in terms of image plane crowd density on the **Campus** dataset.

VI. Conclusion

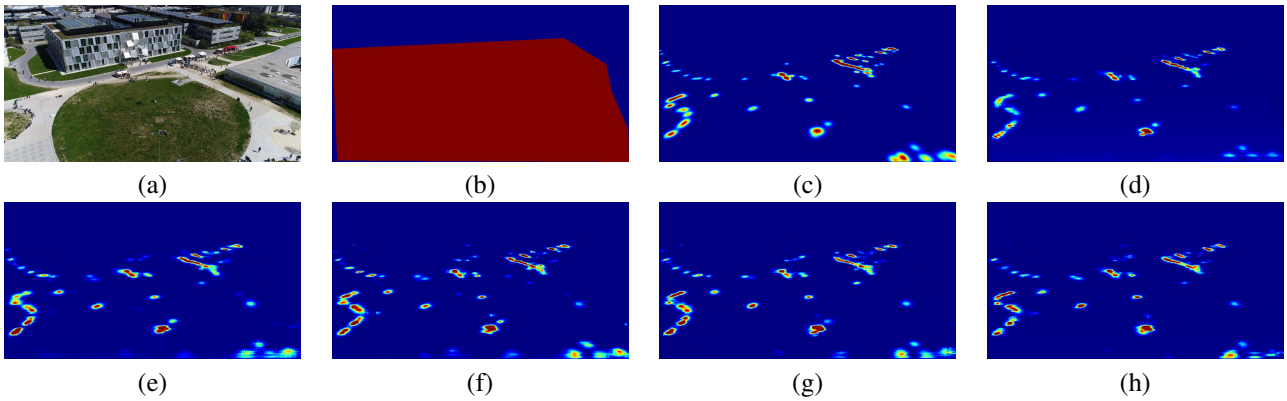
In this paper, we have shown that providing to a deep net an explicit model of perspective distortion effects as an input, along with enforcing physics-based spatio-temporal constraints, substantially increases performance. In particular, it yields not only a more accurate people count but also a better localization of the high-density areas, as can be seen in Fig. 4.

Model	MAE	$RMSE$	$MPAE$
CSRNet [11]	38.5	42.7	121.3
MCNN [4]	132.7	145.3	367.6
SwitchCNN [6]	61.2	72.9	163.2
OURS-NoGeom	36.8	39.9	115.7
OURS-GeomOnly	26.1	35.3	107.2
OURS (frame interval 1)	24.8	32.7	103.2
OURS (frame interval 5)	**18.2**	**26.6**	98.7
OURS (frame interval 10)	22.9	34.3	**94.2**

TABLE III: Comparative results in terms of head-plane crowd density on the **Venice** dataset.

Model	MAE	$RMSE$	$MPAE$
CSRNet [11]	39.2	44.0	124.7
MCNN [4]	133.7	148.4	368.2
SwitchCNN [6]	63.1	75.8	165.4
OURS-NoGeom	37.2	40.4	116.3
OURS-GeomOnly	27.3	37.2	108.9
OURS (frame interval 1)	25.2	33.4	104.7
OURS (frame interval 5)	**18.7**	**27.0**	99.2
OURS (frame interval 10)	23.6	35.2	**95.1**

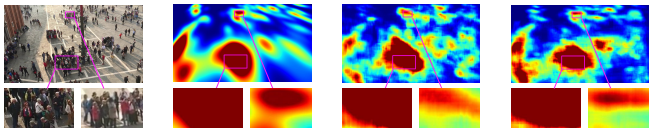
TABLE IV: Comparative results in terms of image plane crowd density on the **Venice** dataset.



(a) (b) (c) (d)

(e) (f) (g) (h)

Fig. 3: **Crowd density estimation on the Campus dataset. (a)** Input image. **(b)** ROI overlaid in red. **(c)** Ground truth head plane density. **(d-h)** Density maps generated by **OURS-NoGeom**, **OURS-GeomOnly**, **OURS**(1), **OURS**(5), and **OURS**(10).



(a) (b) (c) (d)

Fig. 4: **Crowd density estimation on the Venice dataset. (a)** An image of Piazza San Marco in Venice. The two purple boxes highlight patches in which the crowd density per square meter is similar. **(b)** Ground-truth *head plane density*. **(c)** Density maps generated by **OURS-NoGeom**. **(d)** Density maps generated by **OURS**(5). Note that the peak densities are very close, whereas they are quite different in the **OURS-NoGeom** version.

This is of particular interest for crowd counting from a moving drone that can register its camera with respect to the scene using its internal sensors and therefore estimate the required perspective model. Our approach is equally applicable to mobile device that also possess internal sensors or can use standard photogrammetric techniques to estimate their 3D pose. In future work, we will incorporate this approach into the landing system of the drone to allow for automated landing in potentially crowded scenes.

REFERENCES

[1] V. Lempitsky and A. Zisserman, "Learning to Count Objects in Images," in *Advances in Neural Information Processing Systems*, 2010.

[2] A. Chan and N. Vasconcelos, "Bayesian Poisson Regression for Crowd Counting," in *International Conference on Computer Vision*, 2009, pp. 545–551.

[3] C. Zhang, H. Li, X. Wang, and X. Yang, "Cross-Scene Crowd Counting via Deep Convolutional Neural Networks," in *Conference on Computer Vision and Pattern Recognition*, 2015, pp. 833–841.

[4] Y. Zhang, D. Zhou, S. Chen, S. Gao, and Y. Ma, "Single-Image Crowd Counting via Multi-Column Convolutional Neural Network," in *Conference on Computer Vision and Pattern Recognition*, 2016, pp. 589–597.

[5] D. Onoro-Rubio and R. López-Sastre, "Towards Perspective-Free Object Counting with Deep Learning," in *European Conference on Computer Vision*, 2016, pp. 615–629.

[6] D. Sam, S. Surya, and R. Babu, "Switching Convolutional Neural Network for Crowd Counting," in *Conference on Computer Vision and Pattern Recognition*, 2017, p. 6.

[7] F. Xiong, X. Shi, and D. Yeung, "Spatiotemporal Modeling for Crowd Counting in Videos," in *International Conference on Computer Vision*, 2017, pp. 5161–5169.

[8] V. Sindagi and V. Patel, "Generating High-Quality Crowd Density Maps Using Contextual Pyramid CNNs," in *International Conference on Computer Vision*, 2017, pp. 1879–1888.

[9] Z. Shen, Y. Xu, B. Ni, M. Wang, J. Hu, and X. Yang, "Crowd Counting via Adversarial Cross-Scale Consistency Pursuit," in *Conference on Computer Vision and Pattern Recognition*, 2018.

[10] J. Liu, C. Gao, D. Meng, and A. Hauptmann1, "Decidenet: Counting Varying Density Crowds through Attention Guided Detection and Density Estimation," in *Conference on Computer Vision and Pattern Recognition*, 2018.

[11] Y. Li, X. Zhang, and D. Chen, "CSRNet: Dilated Convolutional Neural Networks for Understanding the Highly Congested Scenes," in *Conference on Computer Vision and Pattern Recognition*, 2018.

[12] D. Sam, N. Sajjan, R. Babu, and M. Srinivasan, "Divide and Grow: Capturing Huge Diversity in Crowd Images with Incrementally Growing CNN," in *Conference on Computer Vision and Pattern Recognition*, 2018.

[13] Z. Shi, L. Zhang, Y. Liu, and X. Cao, "Crowd Counting with Deep Negative Correlation Learning," in *Conference on Computer Vision and Pattern Recognition*, 2018.

[14] X. Liu, J. Weijer, and A. Bagdanov, "Leveraging Unlabeled Data for Crowd Counting by Learning to Rank," in *Conference on Computer Vision and Pattern Recognition*, 2018.

[15] H. Idrees, M. Tayyab, K. Athrey, D. Zhang, S. Al-maadeed, N. Rajpoot, and M. Shah, "Composition Loss for Counting, Density Map Estimation and Localization in Dense Crowds," in *European Conference on Computer Vision*, 2018.

[16] V. Ranjan, H. Le, and M. Hoai, "Iterative Crowd Counting," in *European Conference on Computer Vision*, 2018.

[17] X. Cao, Z. Wang, Y. Zhao, and F. Su, "Scale Aggregation Network for Accurate and Efficient Crowd Counting," in *European Conference on Computer Vision*, 2018.

[18] X. Wang, B. Wang, and L. Zhang, "Airport Detection in Remote Sensing Images Based on Visual Attention," in *International Conference on Neural Information Processing*, 2011.

[19] Z. Lin and L. Davis, "Shape-Based Human Detection and Segmentation via Hierarchical Part-Template Matching," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 4, pp. 604–618, 2010.

[20] C. Arteta., V. Lempitsky, J. Noble, and A. Zisserman, "Interactive Object Counting," in *European Conference on Computer Vision*, 2014.

[21] C. Arteta., V. Lempitsky, and A. Zisserman, "Counting in the Wild," in *European Conference on Computer Vision*, 2016.

[22] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[23] O. Chum and J. Matas, "Planar Affine Rectification from Change of Scale," in *Asian Conference on Computer Vision*, 2010, pp. 347–360.

[24] J. Berclaz, F. Fleuret, E. Türetken, and P. Fua, "Multiple Object Tracking Using K-Shortest Paths Optimization," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 11, pp. 1806–1819, 2011.

[25] W. Liu, M. Salzmann, and P. Fua, "Context-Aware Crowd Counting," in *Conference on Computer Vision and Pattern Recognition*, 2019.