Optimal Camera Placement for 6D Head Pose Estimation

Harshita Soni, Nikhil Tirumala and Aratrik Chattopadhyay
Mercedes Benz R&D India, Bengaluru, India

Keywords: Optimal Camera Placement, Visibility, Binary Integer Programming.

Abstract: Multi-view systems for 6D head pose estimation have applications in human-computer interaction (HCI), virtual reality, 3D reconstruction etc. In a multi-view system, visibility of facial landmarks is essential for accurately regressing 2D landmarks, which are then triangulated to get 3D fiducials. From these 3D fiducials, the 6D head pose is mathematically derived. Optimal camera placement (OCP) is vital for achieving precise pose estimation. OCP can be formulated as a constrained optimization problem that can be solved using Binary Integer Programming. We redefine two key aspects: the visibility criteria and the camera search space. Our visibility algorithm employs a parametric head model to track fiducials, achieving more precise results than ground truth of CMU(Carnegie Mellon University) Panoptic dataset. Additionally, we geometrically optimize the camera search space, deviating from the baseline of uniformly arranged cameras. Through rigorous experimentation, we prove that not only does this refined search space reduce execution time, but also improves the optimality of the solution, giving 99.9% visibility coverage. We also introduce a heuristic method that reduces the constraint-building time from 27 seconds to just 0.07 seconds per control point, while maintaining concise solutions with minimal effects on visibility metrics.

1 INTRODUCTION

6D head pose estimation includes three degrees of freedom for rotation (yaw, pitch, roll) and three for translation (X, Y, Z). Multi-view systems for 6D head pose estimation have various applications across different fields. In AR/VR, head pose estimation enhances the immersive experience by ensuring virtual objects align correctly with the user's perspective. Security applications are equipped with multi-camera surveillance systems where head pose estimation can be used to analyze the behavior and intentions of individuals in surveillance footage. Additionally, such systems can be used for ground truth data collection, like the public dataset, CMU Panoptic [(Joo et al., 2016)], using data captured to train deep learning models.

6D head pose estimation is mathematically derived from 3D landmarks of a set of critical points in a human face, called fiducials, like eye corners, mouth corners, nose-tip, etc. Fiducials form the backbone of most of the downstream functionalities of such a multi-view system. Fiducials are triangulated from their 2D landmarks in multiple camera views. 2D facial landmark regression from an image is a well-known problem that has been addressed using both traditional computer vision techniques and deep learning methods. To summarize, the precision of an end-to-end multi-view system depends on the availability of a certain minimum number of clear image views for the 2D landmark regressor to regress on confidently. Hence, an optimal placement of cameras is essential for the seamless functioning of a multi-view system.

The concept of optimal camera placement (OCP) was outlined by (Chvátal, 1975) in the art gallery problem in 1975. It aims to determine the minimum number of guards (or cameras) needed to cover an entire area.

OCP for surveillance applications does not require tracking objects in 3D. In the case of OCP for 3D objects, in most use cases, the object is static and in a fixed, known position and orientation. In this work, we solve OCP for an application where the target has continuous changes in 6D pose. Also, we have a stringent requirement of having the selected fiducials of the object to be visible in all the 6D poses in at least two cameras. This makes the OCP for a 3D-object-in-motion a more challenging task.

Another aspect that complicates the visibility computation in our case is that the human head is a complex manifold. The visibility of the fiducials is

Soni, H., Tirumala, N. and Chattopadhyay, A.
Optimal Camera Placement for 6D Head Pose Estimation.
DOI: 10.5220/0013715100003982

In *Proceedings of the 22nd International Conference on Informatics in Control, Automation and Robotics (ICINCO 2025) - Volume 1*, pages 82-92
ISBN: 978-989-758-770-2; ISSN: 2184-2809

sensitive to small changes in head position and orientation, leading to self-occlusion.

The OCP can be solved using two main approaches: an iterative method, which is not always optimal, and binary integer programming (BIP), which guarantees an optimal solution.

Finding an optimal solution using BIP is an NP-hard problem. Optimization of any of the inputs to BIP would reduce computation time and allow for the quick arrival of a solution. This is overlooked in most previous works, where they merely discretize the search space into uniform grids. This work shows that initializing the search space geometrically leads to reduced computation time and an optimal solution. We also show that our geometric approach makes the solution generic, which can be adapted to any larger 3D environment.

The end-to-end pipeline could be computationally intensive due to framing and solving the constraints. In this work, we propose an alternate way to precompute and approximate the visibility to save time during execution. To summarize, the contributions of our paper are:

1. We present an algorithm to check the visibility of fiducials and model self-occlusion for a human head.
2. We propose an initialization technique for camera search space that drastically reduces the search space and execution time and optimizes the solution.
3. We propose an approach to reduce the execution time by approximating the visibility.
4. We show that camera solutions obtained from our method can be flexibly fitted to any larger, arbitrarily shaped 3D environment.
5. We show the superiority of our method over the baseline on multiple test metrics.

The paper reviews related work in section 2. Section 3 covers the problem statement, including the visibility model (3.1), OCP formulation (3.2), and camera search space optimization (3.3). Visibility approximation is discussed in section 3.4. Methodologies are evaluated in section 4, with experiments detailed in section 5. The paper concludes with key findings in section 6.

2 RELATED WORK

There has been a lot of work in the domain of optimal camera placement (OCP) after its origin from the art gallery problem [(O'Rourke, 1993)].

(Hörster and Lienhart, 2006) has presented the optimal camera configuration as an integer linear programming problem, which can incorporate different constraints and cost functions pertinent to a particular application. They approximate the continuous camera space by sampling the positions and poses. The visibility model is an essential part of OCP, the definition of which changes based on the application. [(Hörster and Lienhart, 2006), (Bettahar et al., 2014)] defined visibility as the field of view of a camera which is taken to be a 2D fixed-size triangle, while the visibility in (Zhao et al., 2008) is based on the projected length of the tag in the image plane. (Puligandla and Lončarić, 2022) figures the visibility of a control point in a camera by checking its presence in the five planes of the FoV pyramid of the camera. The visible point analysis technique of (Zhang et al., 2021) is based on a Hidden Point Removal (HPR) approach. Most of these visibility models are applicable for tracking objects or tags, and some have been simplified to 2D. Given our use case of 6D head pose estimation, tracking of 3D facial landmarks cannot be done by 2D-based projection methods. In contrast, a 3D-based method like HPR is applicable but susceptible to misclassification errors around regions of high local curvature.

[(Zhao et al., 2008), (Liu et al., 2014), (Bettahar et al., 2014), (Puligandla and Lončarić, 2022)] uniformly divide the camera configuration space into grids and populate the camera search space by placing cameras in those grids. (Zhang et al., 2015) adopts the technique of local optimization of a single camera, followed by iterative addition of cameras to capture the uncovered surfaces, not ensuring optimality of the overall solution. (Zhang et al., 2021) employs a genetic algorithm for global optimization of camera configurations.

3 OCP FOR MULTI-VIEW SYSTEM

3.1 Visibility Model

We use an off-the-shelf 3D head parametric model that defines a human head and neck with N_v vertices and N_f faces or triangles. The head model is used to model visibility of a fiducial in a camera. All the cameras are assumed to be pinhole. Let $P = \{P_1, P_2, ..., P_{N_v}\}$ be the set of vertices, and $F = \{F_1, F_2, ..., F_{N_f}\}$ be the set of triangles in the head mesh. Fiducial points (N_K) are some critical points subsetted from P. Let $K = \{k_1, k_2, ..., k_{N_K}\}$

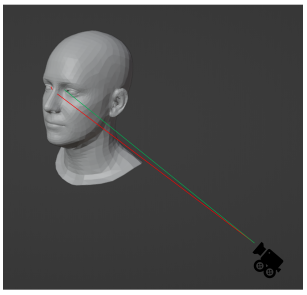

Figure 1: Visibility of a fiducial in camera. Left eye inner corner is visible in camera while right eye inner corner is obstructed by nose.

be the indices of the fiducial points in P. Each triangle F_i is constructed from 3 vertices given by $vertex\_to\_face(P_{i_1}, P_{i_2}, P_{i_3})$ where $i_1, i_2, i_3 \in [1..N_v]$. Let C_j be a camera in multi-view system. A 3D point P_k is said to be visible in C_j if

1. P_k is in the field-of-view (FoV) of C_j and
2. P_k is not occluded by any of the triangles in F when viewed in C_j 1

We model the self-occlusion of fiducial P_k as an aggregation of the intersection of $\vec{C_jP_k}$ with all the triangles of the mesh. We check the intersections using the concept of barycentric coordinates.

The self-occlusion of a fiducial depends on the 6D pose of the head. A 6D pose, Q_i consists of 3 degrees of freedom for rotation R (yaw, pitch, roll) and 3 degrees of freedom for translation T (X, Y, Z). The flame mesh F is obtained by transforming a neutral posed mesh N by a 6D pose Q_i.

Let $Intersection(P_k, F_i, C_j)$ be a binary flag representing the intersection status of $\vec{C_jP_k}$ with triangle F_i. The intersection status of P_k with F_i in the triangle, models self-occlusion and is given by -

$$self\_occl(P_k, Q_i, C_j) = \bigwedge_{i=1}^{N_f} Intersection(P_k, F_i, C_j), \quad \forall F_i \in F - \{vertex\_to\_face(P_{i_1}, P_{i_2}, P_{i_3}) \mid s.t. i_1 \neq k, i_2 \neq k, i_3 \neq k\} \tag{1}$$

The visibility of a fiducial P_k in a head transformed by Q_i from camera C_j is given by

$$Vis(P_k, Q_i, C_j) = self\_occl(P_k, Q_i, C_j) \wedge FoV(F, C_j) \tag{2}$$

where $FoV(F, C_j)$ checks the mesh F projected on camera C_j if it lies on its image plane.

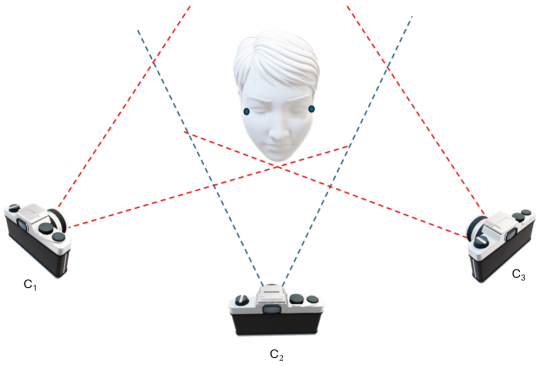

Figure 2: FoV overlap.

3.2 Optimal Camera Placement

Let C denote camera search space of size N_C where $C_j \in C$ is characterized by the 3D spatial location $(X_{C_j}, Y_{C_j}, Z_{C_j})$ and orientation $(yaw_{C_j}, pitch_{C_j})$ of the camera C_j. We define the control space Q as a set of all the head movements we want to capture from multiple views, and eventually constrain the optimization problem on it. $Q_i \in Q$ is composed of $(X_{Q_i}, Y_{Q_i}, Z_{Q_i}, yaw_{Q_i}, pitch_{Q_i}, roll_{Q_i})$ representing a head movement. X is in horizontal axis, Y in vertical axis and Z in depth axis. Control points are the fiducial points in the neutral flame mesh transformed by the control space. A control point can be indexed by 6D pose of the head Q_i and fiducial P_k.

Given the above formulation, the objective is to minimize the number of cameras in the multi-view solution so that each fiducial point is visible in at least 2 cameras. Hence, the camera placement problem can be formulated as -

$$Minimize \sum_{j=1}^{N_C} b_{C_j} \qquad s.t. \tag{3}$$

$$\sum_{j=1}^{N_C} b_{C_j} * Vis(P_k, Q_i, C_j) \geq 2, \forall i = 1, 2, ...N_Q \tag{4}$$

$$\sum_{C_j \text{ at } X,Y,Z} b_{C_j} \leq 1 \tag{5}$$

where b_{C_j} is a binary variable to indicate the membership of camera C_j in the solution. The first constraint eq. 4 is defined for N_K fiducial landmarks. In addition to the visibility constraints, spatial constraints to avoid placing two cameras (in different orientations) at the same 3D spatial location are also imposed. The objective is constrained on a total of $N_Q * N_K$ (visibility) + $|Unique(X_{C_j}, Y_{C_j}, Z_{C_j}) \in C|$ (spatial) constraints. This optimization problem can be solved by binary integer programming (BIP). We show empirically in section 3.3 that problem need

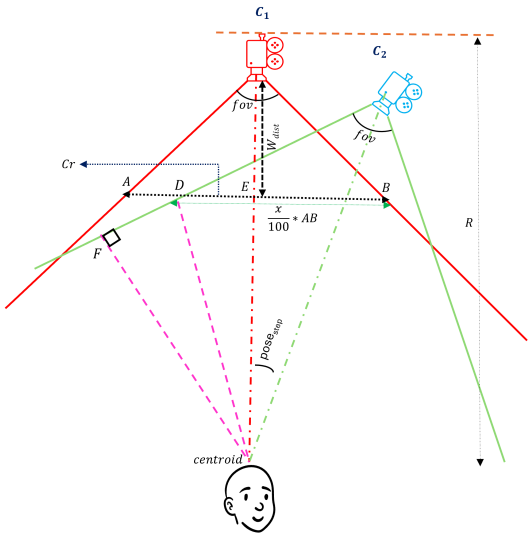

Figure 3: Derivation of step size of yaw in 2D.

need not be constrained on FoV if C is defined as proposed.

3.3 Optimization of Camera Search Space

To geometrically optimize C, we solve for an arrangement of cameras in which adjacent cameras share a minimum degree of overlap between their coverage. This criterion is driven by the need for a minimum of 2 views for triangulation. Spherical placement is the optimal placement which we show empirically in the following sections.

We assume all the candidate cameras are static perspective cameras with the same horizontal and vertical FoV. The camera lens has a minimum and maximum working distance within which the scene can be captured with adequate focus and resolution. The scene captured by any one camera is a frustum of a quadrilateral pyramid.

The cameras can be placed on the surface of a 3D sphere constructed around the centroid of the control space (Q) which is denoted by *centroid*. $(X_{C_j}, Y_{C_j}, Z_{C_j}, Yaw_{C_j}, Pitch_{C_j})$ of C_j are worked out given the sphere's radius (R), the field of view of C_j, and the overlap of coverage desired between adjacent cameras. Given the geometric structure of frustum, ensuring that the head is covered at the nearest depth (Z_{Q_i}) within the control space will automatically guarantee coverage at $Z > Z_{Q_i}$. The critical region (Cr) is the width of the volume that needs to be covered by a camera and is dependent on the use case. Let us consider an $x\%$ overlap in the critical region between adjacent cameras. This means the overlap at greater depths within the control space will be $> x\%$. Regarding the amount of overlap, as shown in fig. 2, C_1 and C_2 can cover the fiducial points on the left side of the head, while C_2 and C_3 can cover the right. A 50% overlap would have sufficed if the head were static. However, to ensure complete coverage of the fiducial points throughout the control space, we opt for an overlap greater than 50%. This approach guarantees ample options for the BIP to select from in the camera search space. We determine the radius R based on the largest sphere that can fit within the 3D environment. A C derived from Cr and R evades the need of explicitly constraining OCP on FoV. Considering all these factors, we derive the $pose_{step}$ in a dimension at which the cameras should be positioned on the sphere, and the same method can be applied to calculate the steps in other dimensions as well. The $pose_{step}$ $(yaw_{step}, pitch_{step})$ can derive yaw_{C_i} and $pitch_{C_i}$ of cameras. $X_{C_i}, Y_{C_i}, Z_{C_i}$ can be easily indexed as a point on the sphere using yaw_{C_i} and $pitch_{C_i}$. Algorithm for building spherical camera search space using calculated steps of yaw_{step} and $pitch_{step}$ is given in 2.

The coverage overlap computation at the critical region of control space is done in 2D for simplification. A sample derivation of $pose_{step}$ required for R radius, $x\%$ overlap on (Cr) in 2D in shown in fig. 3.

In fig 3, we solve for $\angle C_1OC_2$. Here, the critical region (Cr) that must always be covered by C_1, is highlighted by AB. The minimum working distance (W_{dist}) of camera C_1 is given by C_1E and can be derived based on Cr and fov of C_1. In 2D, the coverage overlap between the cameras C_1 and C_2 is represented by BD. The step size $\angle C_1OC_2$ is solved for by setting BD to $x\%$ of AB which boils down to -

$$\begin{aligned}\angle C_1OC_2 = 90 - \frac{fov}{2} \\ - \cos^{-1}\left(\frac{R\cdot\cos\left(\frac{fov}{2}\right)}{\sqrt{\left(\frac{(x-50)\cdot Cr}{100}\right)^2 + (R-W_{dist})^2}}\right) \\ - \tan^{-1}\left(\frac{(x-50)\cdot Cr}{(R-W_{dist})\cdot 100}\right)\end{aligned} \tag{6}$$

where W_{dist} is given as

$$Wdist = \frac{Cr}{2}\cdot\tan(90 - \frac{fov}{2}) \tag{7}$$

Note that the critical region (Cr), radius R and fov are coupled to each other and are not free variables. As (Cr) increases, R may have to be increased to accommodate the more exhaustive coverage.

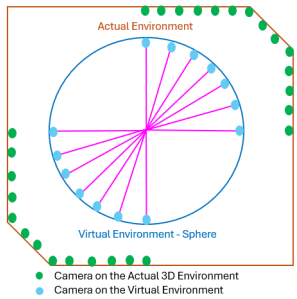

Figure 4: Camera Search Space: Uniform vs Spherical. To cover the same region, the uniform placement requires more cameras than the virtual placement.

3.3.1 Why a Spherical Placement?

When creating the camera search space for an arbitrary-shaped target 3D environment, intuitively, we may place the cameras all over the surface as shown by the outer polygon in fig. 4, encapsulating the region of interest. Naively placing cameras on the surface or dividing the possible camera configuration space into uniformly spaced grids may lead to redundancy of cameras in visibility space and may not ensure sufficient camera options for the BIP.

The bigger the camera search space, the longer the computation time for building the visibility constraints. However, a bigger camera space does not necessarily ensure an optimal solution. This geometric placement of candidate cameras, derived from the visibility requirements, ensures a relatively more optimal solution than the uniform placement.

3.3.2 Generalizability to a Bigger 3D Environment

The camera solution derived from spherically placed cameras is reusable. It can easily adapt to any larger arbitrary 3D environment. A camera can be moved along the line connecting the camera to the *centroid*, which is also the camera's optical axis in this scenario. The camera's depth can be scaled up spatially (camera can be pushed back) from its original 3D location as long as the face is captured at some minimum resolution needed for a good performance of 2D landmark detector. The concept has been qualitatively demonstrated with some examples and mathematical proof for the same is given in appendix 13. We present the following reasons to justify that moving the cameras along their optical axis ensures maximum reusability-

1. The optical axis of all the cameras in the solution intersect at the *centroid*, which is also the center of the sphere. When the camera is scaled along the optical axis:

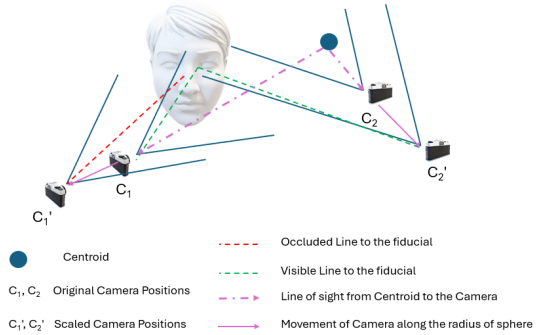

Figure 5: Compensation of visibility on up-scaling of cameras.

   (a) There is no change in the head's visibility located at *centroid*. Only the size of the head in the image changes, and hence it does not affect the visibility of the fiducials.
   (b) When the head is not at the *centroid*, the visibility is affected, but it is least compared to translating the camera in any other direction. In fig. 6 , camera C_1, a part of the camera solution, can see points P_1, P_2 and P_3. Pushing C_1 to C_1' can still see all 3 points. Pushing C_1 to C_2 will favor the visibility of P_1 and P_2, however, will lose P_3. In addition to losing the visibility of P_3, such translations of cameras will have a cascading effect on the remaining cameras in the solution. Computing equivalent transformations/translation of remaining cameras to compensate for the loss of visibility of fiducials is non-trivial. It is as bad as solving the binary integer program all over again. A generalizable method, like ours, must preserve the constraints satisfied by the original camera network as much as possible.
2. Moreover, a single camera's coverage might be affected by the upscaling along optical axis. However, combined coverage of the original solution from all the cameras is maintained in the upscaled solution. For example, in fig. 5, the left eye inner corner was visible in C_1 which upon scaling C_1' is now unable to capture it due to obstruction by nasian bridge. On the other hand, the same fiducial was originally out of FoV of C_2, and is now captured by C_2'.

3.4 Approximate Visibility

Building visibility constraints for a Q_i requires computing $Vis(*, Q_i, C_j)$ with respect to all C_j in C, which is time-consuming and computationally intensive. We propose that visibility need not be computed for points at all depths and from all cameras. The visi-

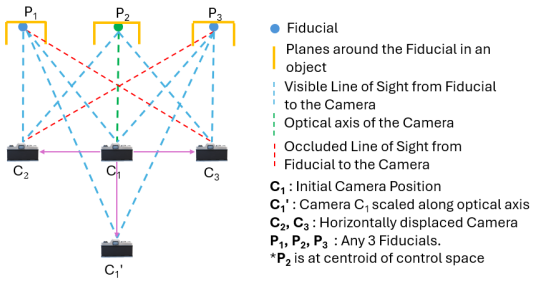

Figure 6: Visibility preservation is maximal for a translation along the camera's line of sight.

bility flags of the fiducial points can be precomputed on a wide range of head positions and orientations from one camera and at one depth (Z_{fixed}), and looked up while building visibility constraints with respect to the other cameras on the fly.

Visibility is unaffected when $P'_k = \alpha * P_k$, meaning a fiducial P_k is moved along the camera's line of sight by a scale of α. If any other point P_l can be expressed as a scaled version of P_k, the visibility of P_l will be the same as that of P_k. Readers may refer to the appendix 13 for proof.

Every 3D point can be represented as a scaled version of a point at a fixed depth Z_{fixed}. Hence, the visibility of control points need not be computed at all depths, as long as we have the visibility flag of a scaled version of P_k at Z_{fixed} available. If the exact match is unavailable in the visibility lookup table LUT_{vis}, the visibility flag of the nearest control point is used. This approach can be used in cases where a certain degree of approximation is acceptable to trade off with the runtime computation. To reduce the approximation, LUT_{vis} must be populated with a finer granularity and broader ranges of head positions (X,Y) and orientations (yaw, $pitch$, and $roll$).

A LUT_{vis} is populated with visibility flags for N_K fiducials at $(|yaw\_ranges| * |pitch\_ranges| * |roll\_ranges|)$ head orientations and $(|X\_ranges| * |Y\_ranges|$ head positions computed at $Z = Z_{fixed}$. A step-by-step algorithm for obtaining (approximate) visibility of P_k from C_j given a LUT_{vis} is shown in algorithm 1.

4 EVALUATION

To evaluate the effectiveness of our approach, we define the metrics as follows -

1. Conciseness(N_S): The fewer cameras in the solution, the more concise the multi-view solution.
2. Test Visibility Metric(η): The control points that failed to be captured by two or more cameras are categorized as **Failure**. Test visibility is defined as -

$$\eta = 1 - \frac{\text{Failure}}{N_Q^{\text{test}}} \tag{8}$$

where N_Q^{test} is the size of test control space.

3. Constraint Building Time (T_{vis}) : Time complexity for building visibility constraints is $O(N_Q * N_C * N_K)$ as there are N_Q visibility constraints for each of the N_K fiducial points. Building each constraint requires visibility computation from each of N_C cameras.
4. Camera Exposure Rate (β): We define the camera exposure rate of a fiducial point as the average number of cameras in the multi-view solution that can capture the fiducial point.

Algorithm 1: Algorithm depicting the proposed approximate visibility computation of P_k from camera C_j.

Input: Visibility Lookup
$LUT_{vis}[yaw\_ranges, pitch\_ranges, roll\_ranges, X\_ranges, Y\_ranges]$
Control Point
$P_k : yaw_p, pitch_p, roll_p, X_p, Y_p, Z_p$
Camera $C_j : yaw_c, pitch_c, X_c, Y_c, Z_c$
Depth Z_{fixed}
Rotation matrix to euler angles : `euler()`
Euler angles to rotation matrix : `rot_mtx()`
Output: Approximate visibility of P_k wrt C_j
$\mathbf{t_p} = [X_p, Y_p, Z_p]$
$\mathbf{t_c} = [X_c, Y_c, Z_c]$
$\mathbf{R_p} = \texttt{rot\_mtx}(yaw_p, pitch_p, roll_p)$
$\mathbf{R_c} = \texttt{rot\_mtx}(yaw_c, pitch_c, 0)$
$\mathbf{R_{p,c}} = \mathbf{R_p}.\mathbf{R_c}^T$
$yaw_{p,c}, pitch_{p,c}, roll_{p,c} = \texttt{euler}(\mathbf{R_{p,c}})$
$\mathbf{t_{p,c}} = (\mathbf{t_p} - \mathbf{t_c}).\mathbf{R_c}$
$X_{p,c}, Y_{p,c}, Z_{p,c} = \mathbf{t_{p,c}}$
$X'_{p,c} = (Z_{fixed}/Z_{p,c}) * X_{p,c}$
$Y'_{p,c} = (Z_{fixed}/Z_{p,c}) * Y_{p,c}$
$yaw' = \underset{y_i \in yaw\_ranges}{\arg\min} |y_i - yaw_{p,c}|$
$pitch' = \underset{p_i \in pitch\_ranges}{\arg\min} |p_i - pitch_{p,c}|$
$roll' = \underset{r_i \in roll\_ranges}{\arg\min} |r_i - roll_{p,c}|$
$X' = \underset{X_i \in X\_ranges}{\arg\min} |X_i - X_{p,c}|$
$Y' = \underset{Y_i \in Y\_ranges}{\arg\min} |Y_i - Y_{p,c}|$
$vis = LUT_{vis}[yaw'][pitch'][roll'][X'][Y']$
return vis;

The mathematical proof showing the correctness of the proposed approximate visibility flags from precomputed visibility is given in the appendix 6.

Table 1: Performance of experiments with different configurations of camera search space.

	Configuration	Radius	N_C	$N_S \downarrow$	$\eta \uparrow$	Failure$\downarrow$	$\beta \uparrow$	T_{vis}(s/iter)$\downarrow$
Baseline	Uniform Placement of Camera	-	5416	15	0.854	111529	3	133
Our Method	Default	500	325	31	0.998	1113	11	12
	Diagonal($20°, 20°$)	500	975	20	0.998	1320	8	27
	Translational(100)	500	975	19	0.998	839	8	27
	Diagonal & Translational($20°, 20°, 100$)	500	1625	18	0.998	1247	7	40
	Default	800	325	10	0.999	477	5	12
	Diagonal($20°, 20°$)	**800**	**975**	**9**	**0.999**	**195**	**5**	**27**
	Translational(200)	800	975	9	0.999	296	5	27
	Diagonal & Translational($20°, 20°, 200$)	800	1625	9	0.999	309	5	40

Figure 7: Example images from CMU Panoptic Dataset. The first row shows visible fiducials according to the dataset. The second row shows the visible fiducials according to our algorithm. Non-visible(incorrect) fiducials are colored red.

5 EXPERIMENTS

5.1 Visibility Algorithm

The head mesh model, FLAME[(Li et al., 2017)] is used to model self-occlusion of fiducials. It has $N_v = 5023$ vertices and $N_f = 9976$ triangles. We show the precision of the visibility algorithm on a public dataset, CMU Panoptic Dataset [(Joo et al., 2016)], a multi-view dataset captured in a 3D environment with 31 HD cameras. It has 3D facial landmarks and corresponding visibility flags from all the cameras. We compare the ground truth visibility flags of the following fiducials from the dataset with visibility computed from our method- Left Eye Inner Corner, Left Eye Outer Corner, Right Eye Inner Corner, Right Eye Outer Corner, Left Mouth corner, Right Mouth corner, Nose Tip, Chin. fig. 7 shows the qualitative results.

5.2 Spherical Placement of Candidate Cameras

Coordinate System: In our experiments, we have assumed the X-axis to the right, Y-axis downwards, Z-axis into the screen.

We solve the problem using a small control space of size $N_Q = \mathbf{768}$ and test the solution on a 3X finely sampled control space of size N_Q^{test} **96000**. The 6D ranges of runtime control space and test control space are defined in Table 2. As for fiducial points, we constrain OCP on the visibility of all the fiducials mentioned in section 5.1. We do not add FoV constraints (eq. 4) as they are implicitly satisfied by all the candidate cameras. All experiments use an open-source CBC solver[(Forrest et al., 2024)].

Baseline: Assuming the scene is set in a rectangular room, we imitate the baseline by placing 5416 cameras in uniform grids on the front and side walls, with the person's head directed towards the front wall.

Our Method: Camera search space is created as described in section 3.3. In 6, the horizontal and vertical fov is set to 90° ,hence, $yaw_{step} = pitch_{step}$. We present results for two radii: 500mm and 800mm. For the experiments with $R = 500mm$, the Cr is set to 200mm with an overlap of $x = 60\%$, resulting in a $pose_{step}$ of 14.11°. For the $R = 800mm$, the Cr is set to 400mm with an overlap of $x = 75\%$, yielding $pose_{step}$ of 13.96°. In both cases, the step sizes are approximated to 15°. As stated earlier, Cr, R and fov are interrelated, implying that the geometry in fig. 3 may change if the variables are geometrically inconsistent. For instance, targeting a Cr of 400mm with radius of 500mm will modify the geometry of fig. 3 and in turn the eq. 6. Experiments with optimized search space are performed in 4 different augmentation settings -

1. Default Configuration - Placing the cameras over the sphere at steps of 15° yaw and pitch.
2. Diagonal Augmentation (aug_{yaw}, aug_{pitch}) -
   (a) Default Configuration

(b) 2 additional cameras oriented at aug_{yaw}, aug_{pitch}
- $yaw_{C_j} + aug_{yaw}, pitch_{C_j} + aug_{pitch}$
- $yaw_{C_j} - aug_{yaw}, pitch_{C_j} - aug_{pitch}$

3. Translational Augmentation (aug_{trans}) -
(a) Default Configuration
(b) 2 additional cameras translated at
- $X_{C_j} + aug_{trans}$
- $X_{C_j} - aug_{trans}$

4. Diagonal & Translational Augmentation ($aug_{yaw}, aug_{pitch}, aug_{trans}$) -
(a) Default Configuration
(b) 2 additional cameras rotated and translated at
- $yaw_{C_j} + aug_{yaw}, pitch_{C_j} + aug_{pitch}$
- $yaw_{C_j} - aug_{yaw}, pitch_{C_j} - aug_{pitch}$
- $X_{C_j} + aug_{trans}$
- $Y_{C_j} - aug_{trans}$

In Setting 2, the default configuration is enhanced by orienting the cameras diagonally at specific angles from their original poses. In Setting 3, the default setup is enhanced by cameras translated along a specified dimension. These augmentations aim to provide BIP with more options, if needed. The results from these augmented settings closely resemble those of the default configuration, highlighting the effectiveness of our approach. These augmentations represent a trade-off between the conciseness of the solution and the computational complexity of the search space, allowing users to choose based on their specific use case.

Our extensive experiments demonstrate that the multi-view solution achieved through optimized initialization of the camera search space outperforms the solution obtained from uniform camera placement across all test metrics. The results in Table 1 indicate that our method, with or without augmentation, consistently achieves a high test visibility score (over 99%) while utilizing a much smaller camera search space than the baseline.

A more concise camera search space also leads to significantly reduced execution times compared to the baseline. A low test visibility score is associated with a poor-quality solution as it comes with a higher failure rate at intermediate control points. Although results from some of our experiments are not as compact as those of the baseline solution, such as ones with $R = 500mm$, solutions are still more desirable due to their higher η, better β, and lower T_{vis}.

Our best outcome, featuring a Diagonal (20, 20) configuration with $R = 800mm$, is visualized in fig. 8. The camera solution offers $\approx 14.5\%$ increment on η with 6 lesser cameras than the baseline solution. On average, our method achieves a β of 5 or higher, compared to the baseline score of 3.

Table 2: Range of runtime and test control space.

Dimension	min	max	runtime step	test step
yaw	-90	60	30	10
pitch	-60	30	30	10
roll	-60	30	30	10
X	0	200	200	50
Y	-400	-200	200	50
Z	300	500	200	50

5.3 Generalizability of Camera Solution

To simulate fitting to an arbitrary-shaped 3D environment, we upscale the cameras from the existing solution by random factors. As seen in Table 3, the count of test control points failing to be tracked by the environment-adapted solution ($Failure_{adapted}$) is less than or equal to failures of the original solution ($Failure_{original}$).

The experimental results in Table 1 and 3 highlight the optimality of spherical placement strategy and adaptability of the solutions, respectively.

Table 3: Performance of environment-adapted camera solution vs original solution.

$Failure_{original} \downarrow$	Scale Range	$Failure_{adapted} \downarrow$
1113	1.5-2.0	0
1247	1.0-1.5	121
195	1.2 - 1.6	61

5.4 Camera Placement from Proposed Approximated Visibility

To test our proposed approximated visibility as explained in 3.4, we experiment with 2 LUT_{vis} of different granularities. Both the lookups are populated with visibility flags of the required fiducials in the range of: yaw $[-180°, 180°]$, pitch $[-90°, 90°]$, roll $[-180°, 180°]$ at steps of 10°, 10°, 10°, respectively, and X and Y in range of $[-700, 500]mm$, much bigger than runtime Q [2]. The visibility for the above range of poses is calculated at $Z_{fixed} = 500mm$. The lookup **C-50** has visibility flags stored at spatial granularity (X and Y) of $50mm$, **C-25** at $25mm$. As can be noticed from the results in Table 4, both solutions include a much smaller number of cameras but more invisible test control points than the original solution. The C-50 configuration exhibits greater sparsity in the X and Y dimensions than the C-25 configuration, resulting in the solver using less precise visibility flags than those used in the C-25 configuration. Computationally, the lookup creation can be treated as pre-

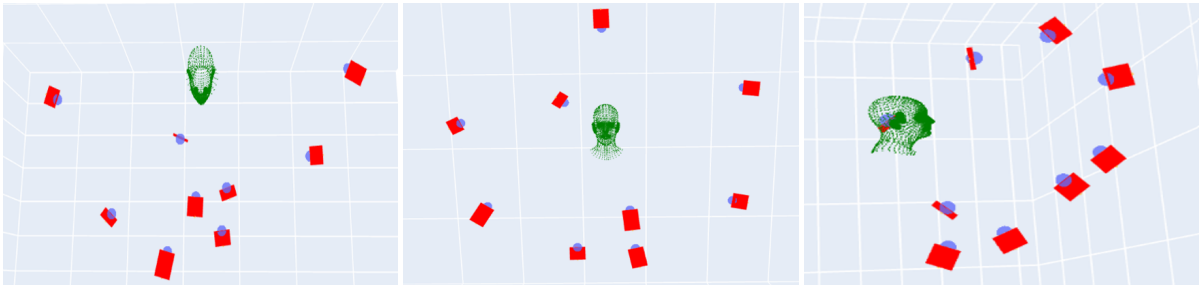

Figure 8: A sample camera solution from our method a) Top View, b) Front View, c) Side View.

processing and utilized for multiple integer programs constrained on various definitions of Q. The trade-off between the performance of a multi-view system and its conciseness highly depends on the nature of the application.

6 CONCLUSION

This work revisits the OCP for 6D head pose estimation where we propose an optimized initialization of camera search space and redefine visibility of 3D points. Furthermore, as an alternative to the computationally intensive task of calculating visibility for all fiducials from every camera, we introduce an algorithm for approximate visibility computation. As a future work, the solution can be extended to be derived from a candidate set of varied focal lengths and PTZ cameras.

Table 4: Performance with Approximated Visibility.

Lookup Configuration	$\eta \uparrow$	$N_S \downarrow$	t_{vis} (s/iter) $\downarrow$
Original	0.998	18	27
C-25	0.878	10	0.07
C-50	0.716	10	0.07

REFERENCES

Bettahar, H., Morsly, Y., and Djouadi, M. S. (2014). Optimal camera placement based resolution requirements for surveillance applications. In *2014 11th International Conference on Informatics in Control, Automation and Robotics (ICINCO)*, volume 01, pages 252–258.

Chvátal, V. (1975). A combinatorial theorem in plane geometry. *Journal of Combinatorial Theory, Series B*, 18(1):39–41.

Forrest, J., Ralphs, T., Vigerske, S., Santos, H. G., Forrest, J., Hafer, L., Kristjansson, B., jpfasano, EdwinStraver, Jan-Willem, Lubin, M., rlougee, a andre, jpgoncal1, Brito, S., h-i gassmann, Cristina, Saltzman, M., tosttost, Pitrus, B., MATSUSHIMA, F., Vossler, P., SWGY, R. ., and to st (2024). coin-or/cbc: Release releases/2.10.12.

Hörster, E. and Lienhart, R. (2006). On the optimal placement of multiple visual sensors. In *Proceedings of the 4th ACM International Workshop on Video Surveillance and Sensor Networks*, VSSN '06, page 111–120, New York, NY, USA. Association for Computing Machinery.

Joo, H., Simon, T., Li, X., Liu, H., Tan, L., Gui, L., Banerjee, S., Godisart, T., Nabbe, B. C., Matthews, I. A., Kanade, T., Nobuhara, S., and Sheikh, Y. (2016). Panoptic studio: A massively multiview system for social interaction capture. *CoRR*, abs/1612.03153.

Li, T., Bolkart, T., Black, M. J., Li, H., and Romero, J. (2017). Learning a model of facial shape and expression from 4d scans. *ACM Trans. Graph.*, 36(6).

Liu, J., Sridharan, S., Fookes, C., and Wark, T. (2014). Optimal camera planning under versatile user constraints in multi-camera image processing systems. *IEEE Transactions on Image Processing*, 23(1):171–184.

O'Rourke, J. (1993). On the rectilinear art gallery problem. *Computational Geometry*, 3(1):53–58.

Puligandla, V. A. and Lončarić, S. (2022). A multiresolution approach for large real-world camera placement optimization problems. *IEEE Access*, 10:61601–61616.

Zhang, H., Eastwood, J., Isa, M., Sims-Waterhouse, D., Leach, R., and Piano, S. (2021). Optimisation of camera positions for optical coordinate measurement based on visible point analysis. *Precision Engineering*, 67:178–188.

Zhang, X., Chen, X., Alarcon-Herrera, J. L., and Fang, Y. (2015). 3-d model-based multi-camera deployment: A recursive convex optimization approach. *IEEE/ASME Transactions on Mechatronics*, 20(6):3157–3169.

Zhao, J., Cheung, S.-C., and Nguyen, T. (2008). Optimal camera network configurations for visual tagging. *IEEE Journal of Selected Topics in Signal Processing*, 2(4):464–479.

APPENDIX

Translating the head without any change in rotation changes the view of the object in the camera, affecting the visibility of the fiducial points. Similarly, rotating the object without translating it also affects the visibility of the fiducials. However, there is a special case of translating the object, so the visibility is unaffected. This happens when the object is translated along the camera's line of sight. Given an image I of a head translated at a 3D location $H = [H_x, H_y, H_z]$ in camera C_j, when moved along the camera's line of sight to the new position $H' = \alpha * H$, the corresponding image $I' = \frac{1}{\alpha} * I$ about the 2D projection of H in I.

For proof, let the camera projection matrix and a perspective projection of point X rotated and translated by R and T, respectively, be given as

$$K = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}, proj(X) = K[RX + T] \quad (9)$$

For simplicity, let us assume the translation of the head is done with respect to a central facial point which we address by head center H. In action, all the points in the head are rotated first and then translated. Hence, in neutral pose, the head center will be at $[0,0,0]$, i.e., the origin of the coordinate system. Any amount of rotation applied to the head centre at $[0,0,0]$ will keep the 3D location of the head unchanged at [0,0,0]. Hence,

$$proj(X) = K[RX + H] = proj(H) \quad (10)$$

when X is the head center in neutral pose.

Proof 1 *: When* $H' \rightarrow \alpha * H$, $proj(H) = proj(H')$

Solving LHS,

$$proj(H) = K[H] = \begin{bmatrix} f_x * \frac{H_x}{H_z} + c_x \\ f_y * \frac{H_y}{H_z} + c_y \end{bmatrix} \quad (11)$$

Solving RHS,

$$proj(H') = K[\alpha * H] = \begin{bmatrix} f_x * \frac{\alpha * H_x}{\alpha * H_z} + c_x \\ f_y * \frac{\alpha * H_y}{\alpha * H_z} + c_y \end{bmatrix} \quad (12)$$

Proof 2 *: When* $H' \rightarrow \alpha * H$, $I' \rightarrow \frac{1}{\alpha} * I$

Let the facial landmarks be denoted by $P = [P_1, P_2, ..., P_{N_k}]$ where $P_i = [P_x, P_y, P_z]$ (we will work with one sample fiducial point, for simplicity denoted by P). Upon scaling, $H' = \alpha * H$, $P' = P + (\alpha - 1) * H$

Knowing that an image is a set of 2D points, we consider the distance between any two known 2D points to compute the equivalent transformation to be applied to the image.

Proof boils down to,

$$\left\| proj(H') - proj(P') \right\|^2 = \frac{1}{\alpha} \cdot \| proj(H) - proj(P) \|^2 \quad (13)$$

Solving RHS,

$$\begin{aligned} &= \frac{1}{\alpha} \cdot \sqrt{\left(\left(f_x \cdot \frac{H_x}{H_z} + c_x \right) - \left(f_x \cdot \frac{P_x}{P_z} + c_x \right) \right)^2} \\ &\quad + \left(\left(f_y \cdot \frac{H_y}{H_z} + c_y \right) - \left(f_y \cdot \frac{P_y}{P_z} + c_y \right) \right)^2 \\ &= \frac{1}{\alpha} \cdot \sqrt{f_x^2 \cdot \left(\frac{H_x}{H_z} - \frac{P_x}{P_z} \right)^2 + f_y^2 \cdot \left(\frac{H_y}{H_z} - \frac{P_y}{P_z} \right)^2} \end{aligned} \quad (14)$$

Solving LHS,

$$\begin{aligned} proj(P') &= proj(P + H' - H) \\ &= \begin{bmatrix} \frac{f_x(P_x + (\alpha-1)H_x) + c_x(P_z + (\alpha-1)H_z)}{P_z + (\alpha-1)H_z} \\ \frac{f_y(P_y + (\alpha-1)H_y) + c_y(P_z + (\alpha-1)H_z)}{P_z + (\alpha-1)H_z} \end{bmatrix} \end{aligned} \quad (15)$$

To simplify the above equation, we make a realistic assumption of $H_z \approx P_z$, meaning the facial landmarks are approximately at the same depth as the head center H. With this assumption, LHS becomes -

$$\begin{aligned} &= \sqrt{\left(f_x \cdot \frac{H_x}{H_z} + c_x - f_x \cdot \frac{(P_x + (\alpha-1)H_x)}{\alpha H_z} + c_x \right)^2} \\ &\quad + \left(f_y \cdot \frac{H_y}{H_z} + c_y - f_y \cdot \frac{(P_y + (\alpha-1)H_y)}{\alpha H_z} + c_y \right)^2 \\ &= \sqrt{\left(f_x \cdot \frac{H_x}{H_z} - f_x \cdot \frac{(P_x + (\alpha-1)H_x)}{\alpha H_z} \right)^2 + } \\ &\quad \left(f_y \cdot \frac{H_y}{H_z} - f_y \cdot \frac{(P_y + (\alpha-1)H_y)}{\alpha H_z} \right)^2 \\ &= \sqrt{f_x^2 \cdot \left(\frac{H_x}{H_z} - \frac{P_x}{P_z} \right)^2 + f_y^2 \cdot \left(\frac{H_y}{H_z} - \frac{P_y}{P_z} \right)^2} \end{aligned} \quad (16)$$

Algorithm 2: Spherical Placement of Cameras.

Input: yaw_{step}, $pitch_{step}$, R, $centroid$
Output: Camera Search Space C
$centroid_X, centroid_Y, centroid_Z = centroid$;
$C = \{\}$;
$yaw = 0°$;
$pitch = 0°$;
while $yaw \leq 360°$ **do**
  while $pitch \leq 360°$ **do**
    $yaw_{C_j}, pitch_{C_j} = -yaw, -pitch$
    $X_{C_j} =$
     $centroid_X + R \cdot \cos(pitch) \cdot \sin(yaw)$
    $Y_{C_j} = centroid_Y - R \cdot \sin(pitch)$
    $Z_{C_j} =$
     $centroid_Z - R \cdot \cos(pitch) \cdot \cos(yaw)$
    $C_j = (X_{C_j}, Y_{C_j}, Z_{C_j}, yaw_{C_j}, pitch_{C_j})$
    Add C_j in C
    $yaw \leftarrow yaw + yaw_{step}$
    $pitch \leftarrow pitch + pitch_{step}$
  end
end
return C;