MoiréBoard: A Stable, Accurate and Low-cost Camera Tracking Method

CHANG XIAO, Columbia University
CHANGXI ZHENG, Columbia University

Camera tracking is an essential building block in a myriad of HCI applications. For example, commercial VR devices are equipped with dedicated hardware, such as laser-emitting beacon stations, to enable accurate tracking of VR headsets. However, this hardware remains costly. On the other hand, low-cost solutions such as IMU sensors and visual markers exist, but they suffer from large tracking errors. In this work, we bring high accuracy and low cost together to present MoiréBoard, a new 3-DOF camera position tracking method that leverages a seemingly irrelevant visual phenomenon, the moiré effect. Based on a systematic analysis of the moiré effect under camera projection, MoiréBoard requires no power nor camera calibration. It can be easily made at a low cost (e.g., through 3D printing), ready to use with any stock mobile devices with a camera. Its tracking algorithm is computationally efficient, able to run at a high frame rate. Although it is simple to implement, it tracks devices at high accuracy, comparable to the state-of-the-art commercial VR tracking systems.

ACM Reference Format:
Chang Xiao and Changxi Zheng. 2021. MoiréBoard: A Stable, Accurate and Low-cost Camera Tracking Method. 1, 1 (August 2021), 18 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Numerous HCI applications rely on a common building block: tracking a device's spatial location in a fast and accurate way. Perhaps the most demanding use of device tracking is Virtual Reality (VR) and Augmented/Mixed Reality applications. Many commercial VR systems utilize dedicated hardware (e.g., Valve Index [9] and HTC Vive [11]), such as laser and infrared beacon stations, to enable accurate device tracking, but often at high cost.

Other systems such as Google Cardboard aim to offer the user VR experience at low cost. Often repurposing the mobile phone as a VR headset without introducing additional hardware, these systems track the device using the on-device inertial measurement unit (IMU). The IMU sensor provides sufficient accuracy to track the device's orientation [24]—some of the commercial VR devices even solely use IMU sensors to recover device orientation [25]. However, when it comes to position tracking, the IMU sensor falls short. This is because the sensor measures only the device's acceleration, and requires double time integration to recover its position. As the tracking period increases, the time integration suffers from an increasingly larger drifting error.

Another option is to use visual markers [16], such as a checkerboard pattern, displayed on a planar area in the scene. When a camera captures the visual markers, camera location can be recovered from the pixel coordinates of the detected markers. The markers are fully passive, require no power, and cost as little as a piece of paper. Yet, the tracking accuracy is not on par with the active methods (such as the laser-emitting beacon-based solution [8, 9]). When the

Authors' addresses: Chang Xiao, Columbia University, chang@cs.columbia.edu; Changxi Zheng, Columbia University, cxz@cs.columbia.edu.

Manuscript submitted to ACM

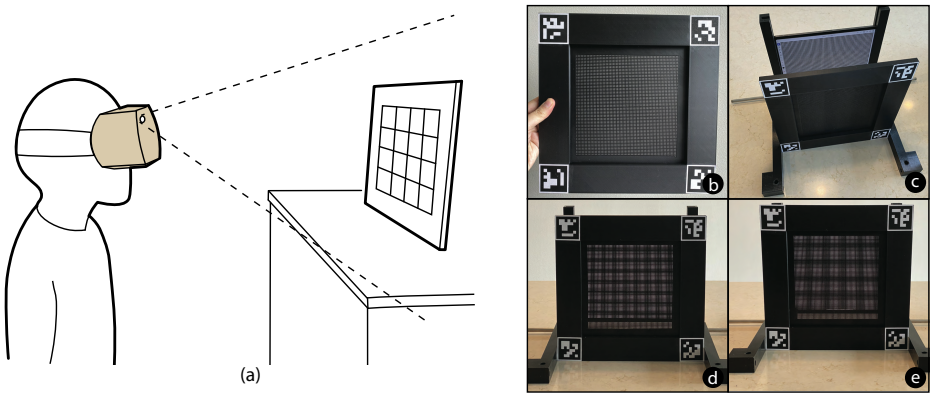

Fig. 1. **MoiréBoard tracking. (a)** A typical application scenario: in a low-cost VR system such as Google Cardboard, MoiréBoard can be used to track camera position and provide tracking accuracy comparable to state-of-the-art VR tracking systems that are much more expensive. **(b)** 3D printed prototype: The total dimension is 30cm×30cm, with a 20cm×20cm fringe pattern area in the center. **(c)** We use an LED display (i.e., an iPad) to display the base layer of gratings, while the front layer is a 3D printed grating structure. The use of LED display is merely for the convenience of experiments (e.g., to test different design parameters). It can be replaced with another 3D printed structure or a printed paper. **(d, e)** When the camera views at different positions, the fringe pattern appears differently, providing clues for accurate position tracking. (Note: This figure may have aliasing artifacts when viewed on a computer screen due to PDF rendering and reduction of the image resolution. We recommend the reader to zoom-in the PDF and view it.)

distance between the camera and visual markers reaches a few meters, the tracking error is on the order of centimeters, too large for many VR applications.

These solutions seem to suggest a performance-cost trade-off that one must take reluctantly. In this work, we challenge this trade-off. We propose a simple visual marker, called MoiréBoard, that enables accurate camera tracking at low cost. Our approach is as accurate as the beacon station system used in high-end VR devices, while maintaining a low cost—the markers can be easily made (e.g., using a 3D printer), and the tracking algorithm is computationally efficient, allowing for tracking at high frame rate (e.g., at 120FPS).

Fig. 2. **Moiré effect.** A fringe pattern emerges when two line gratings with different periods overlap.

Our idea is to harness a seemingly irrelevant visual phenomenon, the *moiré effect* [5]. When two repetitive line gratings with different periods overlap, a bright-and-dark fringe pattern emerges at a larger period due to the interference of the two gratings (see Fig. 2). A remarkable ability of this fringe pattern is the amplification of position changes: a subtle shift of the relative position of the two gratings will cause a notable change of the fringe pattern (see Fig. 1). In our MoiréBoard design, the two gratings stay fixed, but their projections on the camera's image plane change as the camera moves. As a result, the camera captures a fringe pattern whose change can be easily discerned and measured.

Through a systematic analysis of this process, we derive a formula to recover the **3-DOF camera position** from a detected fringe pattern. The camera rotation, if needed by an application, can be recovered by an IMU or sensor fusion algorithms with sufficient accuracy [24]. Unlike other visual-marker-based approaches which require camera calibration to recover intrinsic parameters (such as focal length) before the tracking starts, our formula is calibration-free. Our analysis further informs MoiréBoard's design parameters—such as its physical size and grating periods. Our choice of these parameters is well justified by a theoretical accuracy analysis. Lastly, we prototype MoiréBoard using 3D printing, by which we compare our method with existing tracking methods, and demonstrate its accuracy, performance, and easiness to use.

2 RELATED WORK

Position tracking of VR/AR headsets has been extensively studied in the past few decades. Here we review the most relevant methods to MoiréBoard, namely optical tracking methods, while briefly discussing other methods that emerged recently.

Optical tracking. Currently, most commercial VR devices track headsets and controllers using optical signals. For example, the Valve Index [9] and HTC Vive [11] use the *Lighthouse* tracking system, relying on multiple laser base stations that emit infrared light beams. VR headsets and controllers then use the information of received IR signals to estimate their 3D locations. According to Niehorster et al. [32], this is by far the most accurate tracking system used in commercial VR devices. However, due to the need for dedicated laser hardware, this system is very expensive. Sony PlayStation VR [27] adopts a different solution; the VR headsets and controllers emit visible light, which is received by a standalone camera at a fixed position; the camera then computes the device's position. This approach reduces the cost of the system but is not as accurate as the Lighthouse system. In addition, it may not work well in the -+*presence of strong ambient light, which reduces the signal-to-noise ratio received by the camera.

To avoid the use of external beacon stations (and thereby reduce cost), some VR systems adopt Simultaneous Localization and Mapping (SLAM) [6, 37, 38, 45] for camera tracking, including the Oculus Quest 2 [13], Microsoft HoloLens [30], and Magic Leap One [28]. Relying only on the camera capturing images, the SLAM algorithm aims to recover the scene structure and camera location simultaneously. However, the performance of SLAM largely depends on the scene structure, surface texture, and light condition [38]. Thus, it is not as reliable as beacon-based methods, as some recent studies suggested [21, 34].

Instead of using dedicated commercial VR devices, many projects in both industry [17, 36] and academia [1, 19, 23, 35] aim to reuse existing mobile devices as VR headsets at low cost. The most notable example is Google's Cardboard platform [17]. On mobile devices, IMU [24, 25] sensors are often used to track the device's orientation. But when it comes to tracking the device position, IMU sensors suffer from drifting errors. Mobile VR systems can also use SLAM for position tracking [31]. However, apart from the limitations of the SLAM technique in general, mobile devices often have more restricted computational budget, that further limits the frame rate of SLAM algorithm.

Another potential solution for mobile VR tracking is to use visual markers [10, 14, 16, 33] displayed on an LED screen or printed on a paper. Provided an image capturing visual markers, the tracking system first detects pixel coordinates of the visual markers, whose layout in 3D space is predefined. With this information, the camera position and orientation can be estimated using the *Perspective-n-Point* (PnP) [2, 26] algorithm. This algorithm requires full knowledge of the intrinsic parameters (such as focal length) of a camera, and thus the camera must be pre-calibrated [47]. The estimated camera position is sensitive to detected marker locations. Even a deviation of few pixels would result in centimeter-level camera position error [22].

Our MoiréBoard can be viewed as a visual-marker-based tracking method in general. From this perspective, most relevant to our method is the work by Armstrong et. al [3, 42] called *Moiré Phase Tracking*. The author introduced a visual marker composed of a glass substrate with printed film artwork bonded to both sides. Similarly, Tanaka et. al [39, 40] introduced a visual marker design based on a microlens array. Both methods used the moiré effect—as the camera position changes, the captured marker pattern will also change, offering a clue for camera position estimation. Yet, lacking a systematic analysis of the pattern formation, these methods relies on linear regression for position estimation. Thus, its tracking accuracy is limited. Recently, Banks et. al [4] explored the connection between camera position and its captured moiré pattern using a two-layer line grating design, but they assume the camera is always

facing towards the pattern which is unrealistic in the real applications. There also exist some work using moiré-based tracking method [12, 44]. Instead of using a fixed two-layer pattern, these methods project the second layer from the tracked object itself. However, the resulting systems are much more complicated than the visual-marker-based tracking.

In contrast, while we follow the basic idea of treating MoiréBoard as a visual marker, we provide a full analysis of the geometry of moiré pattern under camera projection, allowing different applications to choose optimized moiré parameters based on our theory. Our method is accurate, comparable to the commercially available Lighthouse tracking system. Besides, it remains fully passive and low-cost. It is also computationally lightweight, allowing for tracking at a high frame rate. All these traits render it suitable for mobile VR applications.

Other tracking methods. Recent years have also witnessed a significant rise in research interests on device tracking with other signal channels like acoustic signals or electromagnetic field. Although this is not directly related to our work, we briefly highlight several recent works in HCI, including CAT [29], SoundTrack [46], Aura [43] and Millisecond [41]. These methods achieves the tracking accuracy of several millimeters, promising for VR tracking. However, they require additional hardware such as a microphone/speaker array or EM signal generator, and they may suffer from drifting errors as the operating time increases [41]. Also, acoustic signals in general are prone to environment noise; they are therefore less reliable than optical signals. So far, almost all commercial VR tracking systems still use optical tracking solutions.

3 CAMERA TRACKING USING MOIRÉ EFFECT

We now present the core idea of MoiréBoard. Our approach is built on careful analysis of how moiré fringes are formed under a camera view (Sec. 3.1∼3.3). Gaining insight from the analysis, we propose the design of a moiré pattern and a computational algorithm (Sec. 3.4). Together, they enable stable and accurate camera motion tracking.

3.1 The Geometry of Moiré Fringes

Moiré fringes from two superposed layers of repetitive structures have been well studied, and a general theory exists [5, 20]. Here to pave the way toward our core analysis (in Sec. 3.2), we introduce the geometry of a special type of moiré fringes.

The moiré fringes we consider here emerge from two superposed grid structures, which we refer to as grid A and grid B, respectively. Each grid can be viewed as a combination of two perpendicular line gratings (along x- and y-direction, respectively), and let the grating periods of grid A and grid B be T_a and T_b (see Fig. 3). Here we assume $T_\mathrm{a} > T_\mathrm{b}$ without loss of generality. When the two grids superpose on each other, the interference between the gratings forms another grid pattern, generally known as the moiré *fringes*. It has been shown that the period T_m of the fringes (i.e., the grid cell size in Fig. 3a) depends on T_a and T_b in the following way,

$$T_\mathrm{m} = \frac{T_\mathrm{a} T_\mathrm{b}}{T_\mathrm{a} - T_\mathrm{b}}. \tag{1}$$

This relation reveals an interesting property of moiré fringes: when T_a and T_b are close, $T_\mathrm{a} - T_\mathrm{b}$ is so small that the fringe period T_m is much larger than T_a and T_b (due to the division in (1)). Remarkably, this property leads to an amplification of the relative motion between grid A and B, described as follows.

Suppose grid B is translated by its period T_b while grid A stays fixed. After the translation, grid B appears the same as before due to its periodicity. Thus, the moiré fringes must be shifted by an entire period T_m. Formally, a small shift Δx_b of grid B will cause a large movement of the moiré fringes by the amount Δx_m, and the amplification ratio of their

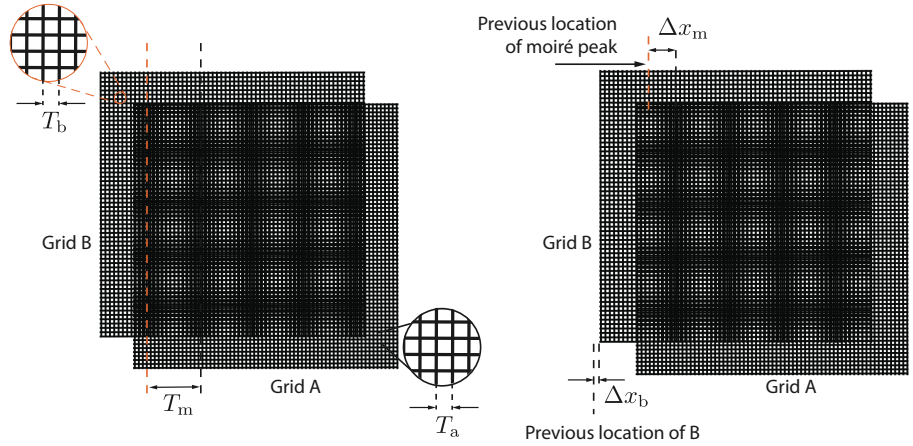

Fig. 3. **Moiré fringes.** (**left**) Two grid patterns with grid size T_a and T_b are overlapped. The resulting moiré pattern is another grid pattern with grid size T_m. (**right**) Grid B is moved by Δx_b while grid A stays fixed. This causes the moiré pattern to be shifted by Δx_m, an amount much larger than Δx_b and thus much easier to detect robustly. Note that in this case the moiré fringes along the y-direction stay unchanged. Thereby, we can track camera motion along x- and y-directions separately.

motions is

$$\frac{\Delta x_m}{\Delta x_b} = \frac{T_m}{T_b} = \frac{T_a}{T_a - T_b}. \tag{2}$$

Inspired by Eqs. (1) and (2), our idea is to harness moiré fringes as an amplifier lens, from which we can track even the subtle motion of grid B with high accuracy. The next step is to bridge the shift of grid B with camera motion so that we can track the camera.

3.2 Moiré Pattern under Camera Projection

We now examine moiré fringes under camera projection and motion—an analysis that to our knowledge has *not* been explored. The goal is to reveal how the camera motion can be recovered through the camera captured moiré fringes.

The camera projects a 3D point $\boldsymbol{X}$ onto a 2D point $\boldsymbol{x}$ on the image plane. In computer vision [2], this projection is expressed as $\boldsymbol{x} = \mathbf{K}\,[\mathbf{R}|\boldsymbol{t}]\,\boldsymbol{X}$, where $\boldsymbol{X} = (x, y, z, 1)^T$ is a 3D point expressed by its homogeneous coordinate in the world frame of reference; $\boldsymbol{x} = (x, y, 1)^T$ is the 2D homogeneous coordinate of the projected point on the image plane. $\boldsymbol{X}$ is first transformed into the camera's local frame of reference by a 3×4 matrix $[\mathbf{R}|\boldsymbol{t}]$, and then projected by a 3×3 matrix $\mathbf{K}$. The transformation matrix $[\mathbf{R}|\boldsymbol{t}]$, known as the camera's *extrinsic* matrix, depends on the camera's orientation and location in 3D space. The $\mathbf{K}$ matrix, known as the *intrinsic* matrix, encodes the camera's focal lengths and optical center, namely,

$$\mathbf{K} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}, \tag{3}$$

where f_x and f_y are the focal lengths along x- and y-direction of the image plane, respectively; and (c_x, c_y) indicates the camera's optical center.

In this work, we aim to recover camera position $\boldsymbol{t}$ given a moiré fringe image, without the knowledge of camera intrinsics (i.e., $\mathbf{K}$). For recovering the rotation $\mathbf{R}$, we use the IMU sensors equipped on mobile devices. This is because IMU sensors offer sufficient accuracy for orientation tracking [25].

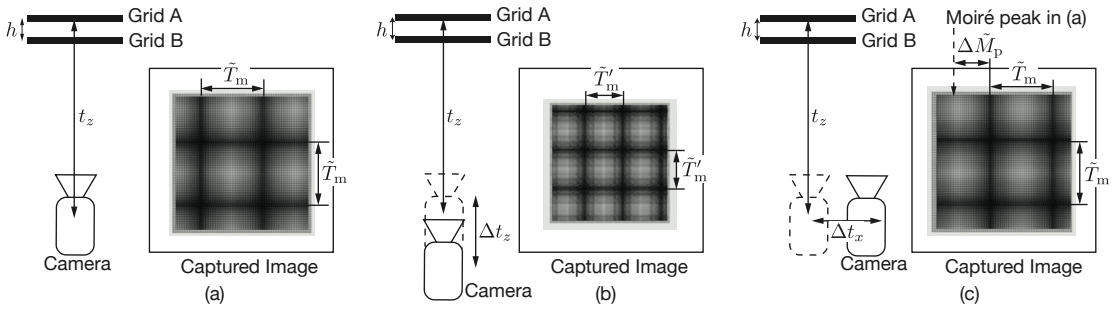

Fig. 4. **Moiré fringes under camera projection.** In each subfigure, the left is a top-down view of the imaging setting, and the right is the captured image. **(a)** The camera looks straight towards the MoiréBoard. The captured moiré fringes have the period $\tilde{T}_\mathrm{m}$. **(b)** When the camera moves further away, the period of captured moiré fringes changes into $\tilde{T}'_\mathrm{m}$. **(c)** When the camera moves horizontally (starting from (a)), the captured moiré fringes are shifted by $\Delta\tilde{x}_\mathrm{m}$, while their period remains unchanged.

To start our analysis, consider two moiré layers: the first one, grid A, is located on the plane $z = 0$, and the second one, grid B, is placed on the plane $z = h$ (see Fig. 4). Assume for now that the camera is located at $(0, 0, t_z)$, looking straight toward the $z = 0$ plane (this assumption will be relaxed shortly). In this case, the camera's rotation matrix remains identity (i.e., $\mathbf{R} = \mathbf{I}$), and the grating period T_a on grid A appears on the captured image with the length $\tilde{T}_\mathrm{a} = T_\mathrm{a}\frac{f}{t_z}$, where f is the camera's focal length (assuming $f_x = f_y = f$ for the simplicity of presentation). Similarly, the projected grating period $\tilde{T}_\mathrm{b}$ of grid B on the captured image is $\tilde{T}_\mathrm{b} = T_\mathrm{b}\frac{f}{t_z - h}$. Applying Eq. (1), we obtain the period of moiré fringes on the captured image (see Fig. 4), namely,

$$\tilde{T}_\mathrm{m} = \frac{T_\mathrm{a}T_\mathrm{b}f}{T_\mathrm{a}h - T_\mathrm{a}t_z + T_\mathrm{b}t_z}. \tag{4}$$

Suppose the camera is translated along x-direction to $(t_x, 0, t_z)$ (see Fig. 4c). Under the camera projection, grid A and B will be translated by different amounts, as they are at different distances from the camera, namely $\Delta\tilde{x}_\mathrm{a} = -t_x\frac{f}{t_z}$ and and $\Delta\tilde{x}_\mathrm{b} = -t_x\frac{f}{t_z - h}$. This amounts to fixing grid A but shifting grid B by $\Delta\tilde{x}'_\mathrm{b} = \Delta\tilde{x}_\mathrm{b} - \Delta\tilde{x}_\mathrm{a}$. According to Eq. (2), the moiré fringes on the captured image will be displaced by

$$\Delta\tilde{x}_\mathrm{m} = \frac{t_x T_\mathrm{a} f}{T_\mathrm{a}h - T_\mathrm{a}t_z + T_\mathrm{b}t_z}. \tag{5}$$

Note that depending on the camera position t_z, the denominators in Eqs. (4) and (5) may be either positive or negative, and so are $\tilde{T}_\mathrm{m}$ and $\Delta\tilde{x}_\mathrm{m}$. A positive $\Delta\tilde{x}_\mathrm{m}$ indicates that the fringes shift toward $+x$ direction on the image plane as the camera moves toward $+x$ direction in space; a negative $\Delta\tilde{x}_\mathrm{m}$ indicates fringes moving opposite to the camera's displacement. Also at a certain t_z, the denominators vanish. We will return to discuss this subtlety in Sec. 3.4.

Equations (4) and (5) lay the foundation of our camera tracking algorithm: in Eq. (4), T_a, T_b, and h are moiré grids' parameters known *a priori*; $\tilde{T}_\mathrm{m}$ can be measured on the captured image. If the focal length f is known, we can solve for t_z. Afterwards, we measure $\Delta\tilde{x}_\mathrm{m}$ from two captured images, and recover t_x using Eq. (5). However, to fully materialize this idea, two questions remain unanswered: (i) how to account for tilted camera (i.e., $\mathbf{R} \neq \mathbf{I}$)? (ii) how to eliminate the need of the focal length f so that our camera tracking is calibration-free? We address both questions next.

3.3 Calibration-free Camera Tracking

Traditional camera tracking, whether marker-based [10, 14, 16] or markerless [7, 31], often require a calibration step to recover camera intrinsics (i.e., $\mathbf{K}$) before tracking starts [18, 47]. Not only does this calibration impose an additional

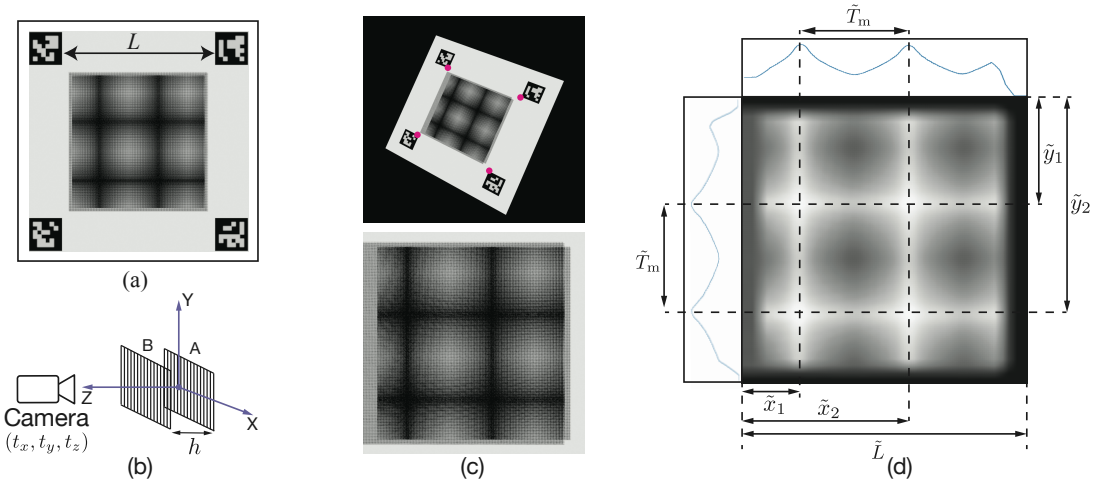

Fig. 5. **Steps of our tracking algorithm. (a)** A MoiréBoard in 2D frontal view. **(b)** Coordinate system of our algorithm. Grid A is located at plane $z = 0$. Grid B is at $z = h$. **(c)** Top: An image of the MoiréBoard captured at a certain camera location. Inner corners of the four visual markers are detected and labeled as red dots. Bottom: We rectify perspective distortion based on the detected corner positions to convert the image into a frontal view image. **(d)** We apply a Gaussian blur filter to the image of (c)-bottom, and then measure $\tilde{T}_\mathrm{m}$, $\tilde{x}_i$ and $\tilde{y}_i$ on the resulting image.

step, it necessitates re-calibration whenever the camera's intrinsic parameters change (such as refocus). Our approach, at first glance, follows the same route, as both Eqs. (4) and (5) require the knowledge of the camera's focal length f.

Elimination of f. A key observation to liberate our approach from calibration is that f simply scales $\tilde{T}_\mathrm{m}$ and $\Delta\tilde{x}_\mathrm{m}$ in Eqs. (4) and (5). Indeed, under camera projection, the physical distance between any two points (on a frontal plane) is scaled by f. In particular, if we place four visual markers on grid A—which are needed anyway to locate moiré fringes in a captured image (see Fig. 5a)—the physical distance L between two markers has an image-plane length $\tilde{L} = f\frac{L}{t_z}$. Since we can easily measure $\tilde{T}_\mathrm{m}$, $\Delta\tilde{x}_\mathrm{m}$, and $\tilde{L}$ from captured images, we take the ratios $\alpha = \tilde{T}_\mathrm{m}/\tilde{L}$ and $\beta_x = \Delta\tilde{x}_\mathrm{m}/\tilde{L}$, which eliminate f from Eqs. (4) and (5). As a result, the camera position coordinate t_z and t_x can be recovered using the following formulas,

$$t_z = \frac{\alpha L T_\mathrm{a} h}{T_\mathrm{a} T_\mathrm{b} + \alpha L\,(T_\mathrm{a} - T_\mathrm{b})} \quad \text{and} \quad t_x = \frac{L}{T_\mathrm{a} h}(\beta_x + n\alpha)(T_\mathrm{a} h - T_\mathrm{a} t_z + T_\mathrm{b} t_z), \tag{6}$$

where n is an integer arising from the moiré fringes' periodicity: two displacements $\Delta\tilde{x}_\mathrm{m}$ and $\Delta\tilde{x}_\mathrm{m} + n\tilde{T}_\mathrm{m}$ lead to the same fringe appearance on the image. In practice, we disambiguate the measurement of $\Delta\tilde{x}_\mathrm{m}$ by exploiting the camera motion's temporal coherence: choose an integer n such that the resulting t_x is closest to t_x from the last camera frame. (for the very first frame, we simply use $n = 0$ and $\beta_x = 0$, which lead to $t_x = 0$). In short, no camera's intrinsic parameters appear in Eq. (6), and thus no camera calibration is needed to recover t_x and t_z. To recover the y-coordinate t_y, we follow the same process as the t_x recovery but measure β_y along the image's y-direction.

Remark. The recovered camera position (t_x, t_y, t_z) is in a world frame of reference, with the unit that we use to measure L (such as millimeter). The world frame of reference is defined as follows. Its x-y plane is the plane of grid A; its z-axis points toward grid B; and its origin is the camera's initial position (at frame 0) projected on the plane of grid A.

Fringes under tilted camera. Our analysis so far assumes that the camera faces straight toward the MoiréBoard's grid plane. When the camera is tilted, the captured fringe pattern becomes distorted, not a square grid anymore. Interestingly, our analysis shown in Fig. 6 suggests a simple way to factor out the effect of camera rotation:

From the captured image, we detect the position of the four visual markers, which are physically located on grid A to indicate the corners of a square region (Fig. 5a). On the image, however, the region indicated by the markers may not be a square due to the tilted camera distortion. But given the image-plane marker positions (i.e., red dots in Fig. 5c), we can construct a homography matrix to rectify the maker positions, restoring the marked region, including its image content, back to a perfect square (see Fig. 5). According to our analysis in Fig. 6, the rectified image is the same as the one captured by a camera at the same location but facing straight to the MoiréBoard's grid plane.

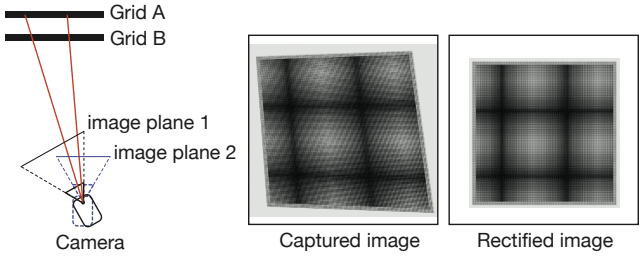

Fig. 6. **Image rectification.** (left) Our key insight is that the moiré fringes (e.g., the peak bands indicated by the red lines) depend only on the positions of grid A and B and the camera location, but not the camera rotation. As the camera tilts, while its image plane has varying orientation, it captures the same set of fringe lines. This is a standard multi-view projection problem in computer vision [2]. One can construct a homography matrix to convert images across different views. In our algorithm, we convert the image captured by a tilted camera (middle) to the frontal view image (right). A mathematical proof is provided in Appendix C.

From there, we can apply Eq. (6) to recover the camera location. To further validate the rotation-invariant properties of MoiréBoard, we also provide a mathematical proof in Appendix C.

Note that the homography transformation can scale the marked region arbitrarily. But our algorithm is agnostic to the image scale, as we only need the dimensionless ratios α and β_x (and β_y) in Eq. (6), not pixel lengths on the image plane. In practice, we always scale the marked region to 1000px×1000px.

Outline of tracking algorithm. We now summarize our camera tracking algorithm:

(1) Capture an image from the camera (Fig. 5c-top).
(2) Detect the four corners of the visual markers (see red dots in Fig. 5c-middle) on the image. In practice, we use ArUco [16] for locating the corners. The side length L of the four corners in physical space is known *a priori*.
(3) Use the detected corners to rectify the captured image to a frontal view image (Fig. 5b-bottom); ensure the size of the marked square region to have 1000×1000 pixels. We then measure $\tilde{L}$, the side length (in pixels) of the four detected corners on the rectified image (Fig. 5d).
(4) Detect peak bands of the captured moiré fringes: First, we convert the rectified image to a grayscale image and invert it (to detect moiré layers' constructive interference bands), followed by applying a Gaussian filter (see Fig. 5d). The Gaussian filter is using a 25x25 kernel with a standard deviation of 4.1 for both horizontal and vertical direction. Next, we average the pixel intensities along x- and y-directions, respectively, reducing the image into two 1D images (see plots in Fig. 5d). From each 1D image, we locate its local peaks. There exist many peak detection algorithms. In practice, we apply a threshold to each 1D image to identify short intervals around the peaks, and then for each interval take the average (1D) pixel position (weighted by pixel intensities).
(5) We measure the following quantities on the 1D images: **(i)** $\tilde{T}_{\mathrm{m}}$, the averaged distance (in pixels) between two consecutive peaks. **(ii)** $\tilde{x}_i$ and $\tilde{y}_j$: the peak locations on the horizontal and vertical 1D images, respectively. Here, the subscripts i and j indicate individual peaks.
(6) For every peak i, compute the displacement of peak locations, namely $\Delta\tilde{x}_{\mathrm{m}}$, with respect to frame 0. This is done by subtracting $\tilde{x}_i$ in frame 0 from its value in the current frame. We then use Eq. (6) to compute t_z and t_x of the

ID	$T_a(mm)$	$T_b(mm)$	$h(cm)$	Working range (m)	Direction
1	3.1	3	10	0.9 to 2.1	S
2	3.1	3	16	1.5 to 3.5	S
3	3.2	3	10	0.6 to 1.3 or 2.1 to 4.0	S or O
4	3.2	3	16	1.0 to 1.7 or 3.3 to 4.0	S or O
5	2.6	2.5	10	0.8 to 1.9	S
6	2.6	2.5	16	1.4 to 3.1	S
7	3.0	2.5	10	0.6 to 1.0	O
8	2.7	2.5	10	0.7 to 1.2 or 1.6 to 4.0	S or O
9	2.7	2.5	20	1.4 to 2.3 or 3.2 to 4.0	S or O

Table 1. **MoiréBoard design parameters.** We list a set of MoiréBoard design parameters and their working ranges. Here we limit the camera's t_z range up to 4m, as it suffices for most VR applications. 'O' indicates that the captured moiré fringes will move in the opposite direction to the camera (due to negative $\Delta\tilde{x}_m$ in (5)), while 'S' indicates that the fringes will move in the same direction as the camera. This table can serve as reference designs when we deploy MoiréBoard in specific applications.

camera location. In this process, we enumerate n in (6) from -5 to 5, and for all possible values of t_x, we choose the one closest to the t_x value of the previous frame.

(7) Meanwhile, we take the detected peak locations $\tilde{y}_j$ along y-direction, and apply the same process as step (6) to recover t_y of the camera position.

3.4 MoiréBoard Design

The geometry of MoiréBoard is specified by its parameters T_a, T_b, and h. We justify the choice of these parameters through closer analysis of Eqs. (4) and (5). First, when the camera is sufficiently far from the MoiréBoard (i.e., t_z is large), Equation (5) leads to a negative $\Delta\tilde{x}_m$, which indicates that the fringes move in a direction opposite to the camera's moving direction. On the contrary, a small t_z leads to a positive $\Delta\tilde{x}_m$ value, and the fringes moves in the same direction as the camera's moving direction. In between, when $t_z = \frac{T_a h}{T_a - T_b}$, the denominators of (4) and (5) vanish. This is the point at which T_a and T_b appear to have the same length under camera projection, and therefore the fringe pattern disappears, as indicated by the fringe period $\tilde{T}_m \to \infty$.

Figure 7 shows the relation of $\alpha = \tilde{T}_m/\tilde{L}$ (i.e., the dimensionless fringe period) with respect to t_z. Although varying widely, $|\alpha|$ must be smaller than 0.5 in practice. Otherwise, the MoiréBoard—which has a finite size $L \times L$—may not be large enough to display two peak bands at the same time, but we need at least two peaks for the measurement of $\tilde{T}_m$ (recall Fig. 5d). Meanwhile, a peak band on the image has certain thickness. Formed by constructive interference of grating lines and due to the use of Gaussian filter (in step (4) of our algorithm), it will never appear as a sharp line. As a result, the fringe period α can not be too small. In practice, we ensure $|\alpha| > 0.1$ to allow for robust measurement of $\tilde{T}_m$.

As illustrated in Fig. 7, when $|\alpha|$ is in the range $[0.1, 0.5]$, the camera motion is restricted in either range 1 (blue) or range 2 (orange). Specific values of these t_z ranges depend on the MoiréBoard's design parameters. In Table 1, we list nine sets of design parameters (i.e., T_a, T_b and h), and their corresponding t_z ranges (for $\alpha \in [0.1, 0.5]$). These parameters are chosen for typical VR applications, wherein the headset moves within a few meters. In practice, one can choose a specific set of parameters depending on their application needs.

3.5 Accuracy Analysis

MoiréBoard is an order of magnitude more accurate than the traditional marker-based tracking methods. We now analyze our approach to understand its benefits, which we will also experimentally confirm in Sec. 4.

All vision-based tracking algorithms, including ours, are designed to map camera displacement to certain measurable image-space changes [2]. For instance, in traditional marker-based approaches (such as [47]), if a camera facing straight to the marker plane undergoes a lateral displacement t_x, then the captured image features will be shifted by $f\frac{t_x}{t_z}$, where f is the camera focal length, and t_z is the distance from the marker to the camera. But using MoiréBoard under the same situation, the captured features (which are moiré fringes) will be shifted by $\Delta\tilde{x}_\mathrm{m}$ defined in (5).

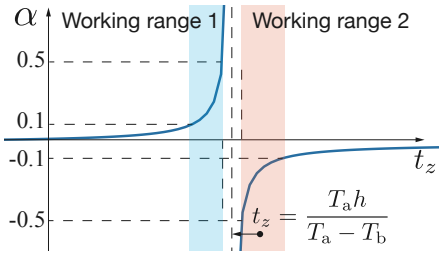

Fig. 7. **The working range of MoiréBoard** is determined by α: (i) we need at least two peak fringes to determine $\tilde{T}_\mathrm{m}$, and this requires $|\alpha| < 0.5$; (ii) $|\alpha|$ must be large enough so that two fringe bands on the image plane can be clearly discerned, and this requires $|\alpha| > 0.1$. To satisfy these constraints, the camera's t_z must locate in either range 1 (blue region) or range 2 (orange region).

To increase the tracking sensitivity, one must enlarge the image-space changes for a given camera displacement. In traditional methods, this demands a larger focal length f (to increase $f\frac{t_x}{t_z}$), but a larger f narrows the camera's field of view, thus limiting the camera's tracking range. In our approach, however, we can enlarge $\Delta\tilde{x}_\mathrm{m}$ by choosing proper T_a, T_b and h without increasing f, and thereby keep the camera's field of view open. Concretely, we can compare the sensitivity of our method with the traditional methods: under the same t_x, t_z and f, the ratio η of the moiré fringe displacement to the image feature displacement in traditional methods is (according to our derivation in Appendix B)

$$\eta = |\Delta\tilde{x}_\mathrm{m}|\frac{t_z}{t_x f} = \frac{|\alpha|L}{T_\mathrm{b}}, \tag{7}$$

This sensitivity boost can be increased by using a larger MoiréBoard size L, and smaller grating period T_b, and it is independent from the distance between the camera and the MoiréBoard. To put it into context, consider a MoiréBoard with $L = 300$mm and $T_\mathrm{b} = 2.5$mm. Recall $|\alpha| \in [0.1, 0.5]$ from Sec. 3.4. According to (7), this MoiréBoard is **12**∼**60**× more sensitive than the traditional methods for tracking camera motion in x- and y-directions. We also analyze the accuracy for tracking in z-direction, and the details are presented in Appendix A.

4 EVALUATION

Prototyping details. There are many ways of making MoiréBoard. We prototype it in a simple way: 3D print the grating structure of grid B with a grating period T_b, and place it in front of an LED display (e.g., we use an iPad). The LED display merely serves as a quick way to display the grating structure of grid A, and can be replaced with other designs such as a piece of paper printed with the grating pattern. We use it for ease of experimenting with different grating structures (e.g., switch to a different T_a). Figure 1 demonstrates the setup and the resulting moiré fringes.

4.1 Synthetic Scene Evaluation

Before evaluating MoiréBoard in real scenes, we test it in synthetic scenes, leveraging the full-fledged image rendering engine that can model camera and scene accurately. We use Blender [15] to set up a MoiréBoard (300mm×300mm) together with four visual markers, each of which has a size 50mm×50mm. We then use the Cycle rendering engine in Blender to synthesize photo-realistic images captured by a virtual camera (see Fig. 8). In this way, we know precisely where the camera is and can compute the locations of the moiré bands on captured images. With this

t_z (m)	Error (pixels)
1.0	0.23 ± 0.14
1.5	0.34 ± 0.17
2.0	0.52 ± 0.31
2.5	0.79 ± 0.44
3.0	0.94 ± 0.57
3.5	1.32 ± 0.61
4.0	2.78 ± 1.79

Table 2. **Peak detection error** in pixels at different t_z.

Fig. 8. **Synthetic images** rendered by Blender Cycle's engine at four different camera positions. We set up a virtual scene and render camera captured images. In this way, we know the ground-truth camera locations, which allows us to evaluate our method's tracking accuracy and compare it with the classic checkerboard-based approach.

ground-truth information, we can evaluate the accuracy at various stages of our algorithm.

Peak detection accuracy. Our algorithm tracks the camera position by measuring image-plane peak locations (recall step (4) of our algorithm in Sec. 3.3). Thus, the tracking accuracy hinges on how accurately we detect the peaks. This motivates us to first examine the peak detection accuracy. To this end, we set up a MoiréBoard with $T_\text{a} = 3.1$mm, $T_\text{b} = 3$mm, and $h = 100$mm. We place the virtual camera at different distances from the MoiréBoard, ranging from $(0, 0, 1)$m to $(0, 0, 4)$m, sampled every 0.5m. At each camera location, we randomly rotate the camera 40 times, while ensuring that from each rotation the visual markers are visible. The camera images are rendered at a resolution of 1920px×1080px. We then compare the detected peak locations with the ground-truth to evaluate the error.

Table 2 reports the peak detection errors at different t_z distances. For each t_z, we compute the average error of peak detection (in pixels) over all 40 camera rotations as well as the error deviation. The results show that the error becomes larger as t_z increases. This is because as the camera moves further away, the MoiréBoard appears in a smaller region on the image. When $t_z < 3$m, the detected peaks always have sub-pixel errors. This experiment suggests that a 300mm×300mm MoiréBoard is suitable for camera tracking within a $\sim$ 4m range, sufficient for most VR applications. For larger tracking ranges, one could use a larger size of MoiréBoard (e.g., a 500mm×500mm MoiréBoard).

Comparison with checkerboard-based tracking. With full knowledge of a scene setup, we can directly compare our method with other vision-based tracking methods. In particular, we consider the checkerboard-based tracking [47], which has become the standard tracking method in numerous applications, and has been implemented in many computer-vision toolboxes such as Matlab and OpenCV.

In this comparison, we use the same MoiréBoard setup as in the previous experiment. In checkerboard-based tracking, we use an 8×8 checkerboard with the physical size 400mm×400mm—the same size as our MoiréBoard plus its optical makers. The tracking algorithm is readily offered in the popular OpenCV library: we use `cv2.findChessboardCorners` to detect the pixel coordinates of checkerboard corners followed by `cv2.solvePnP` to obtain the camera position. The camera's intrinsic parameters (such as its focal length) are known from Blender configuration.

We examine two scenarios: moving the camera along x- and z-directions. In the first scenario, we move the camera along x-direction from $t_x = 0$m to $t_x = 1.4$m, while fixing $t_y = 0$m and $t_z = 2$m. In this process, we sample camera locations every 0.2m. In the second scenario, we move the camera from $t_z = 1$m to $t_z = 3.1$m and sample its position every 0.3m. At every sampled location in both scenarios, we randomly choose 40 camera rotations and recover the camera location from both our method and the checkerboard method. To compare these two methods, we also consider two different image resolutions: *High-res* (1920px×1080px) and *Low-res* (960px×720px).

Figure 9 shows the 3D position errors at each camera position (averaged over 40 random rotations) resulted from both methods. This results indicate that **(i)** the tracking errors of MoiréBoard are an order of magnitude lower than the

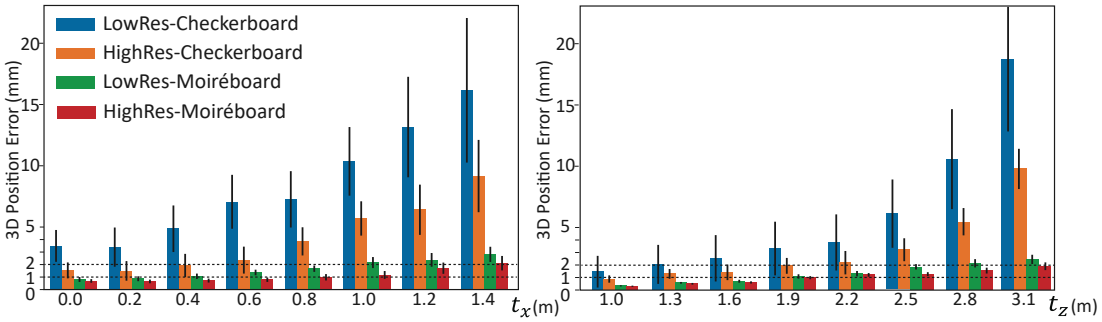

Fig. 9. **Evaluation results on synthetic images. (Left)** Averaged 3D position error at different t_x (t_z=2m). **(Right)** Averaged 3D position error at different t_z. We compare MoiréBoard to the checkerboard-based camera tracking, and test two image resolutions at high-res (1920px×1080px) and low-res (960px×720px). Thanks to the amplification effect of moiré fringes, our method is significantly more accurate than checkerboard. Its performance is also less affected by the image resolution.

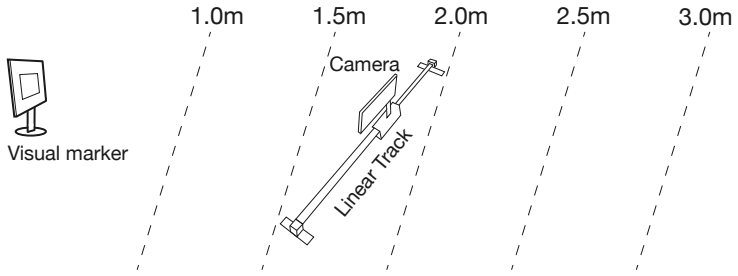

Fig. 10. **Setup of real-scene evaluation.** The visual marker (i.e., MoiréBoard, checkerboard or Lighthouse base stations) is placed at a fixed location. The experiment area is divided into four regions. We repeat the experiments in each region by placing a linear slide that carries the camera moving along a random direction. Without knowing the camera's ground-truth positions, we evaluate the tracking methods by measuring how well they recover the camera's linear motion.

widely used checkerboard-based tracking; and **(ii)** MoiréBoard is much more robust to decreased camera resolution, thanks to its motion amplification through moiré effects.

4.2 Real Scene Evaluation

Next, we conduct experiments in real scenes and compare MoiréBoard to Lighthouse 2.0 [9], the state-of-the-art commercial VR tracking system. Lighthouse 2.0 uses laser-emitting base stations to track the positions of VR headset and controllers. Unlike MoiréBoard, it is not a passive tracking system, and is much more expensive. We also compare our method to the checkerboard tracking method [47], as both are vision-based passive tracking methods.

In real scenes, all tracking methods produce errors, and the ground-truth camera positions are lacking. To measure the tracking errors, we leverage a motorized linear slide, which carries the camera (or VR headset) and translate it linearly in a controlled way. We place the linear slide along different directions. In each run, we collect a series of recovered camera positions and apply linear regression to fit a linear path. The error is measured as the average distance of all recovered position samples to the fitted linear path. In this way, no ground-truth camera positions are needed.

Our experiment setup is illustrated in Fig. 10. We fix the spatial location of MoiréBoard and checkerboard. When using a single base station of Lighthouse 2.0, we place it at the same location as the MoiréBoard and checkerboard.

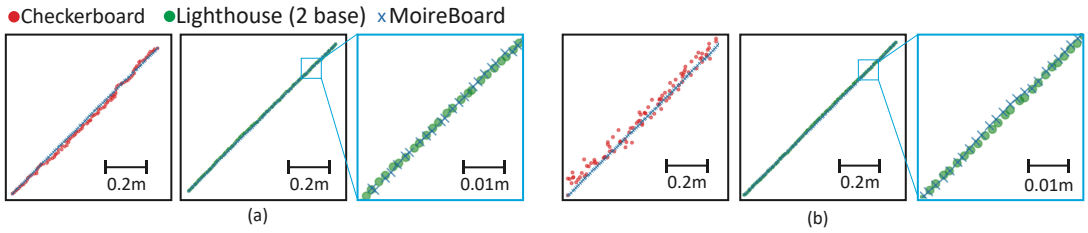

Fig. 11. **Visualization of motion reconstruction.** Here we visualize the x- and z-coordinates of recovered camera positions in several experiments. Individual positions resulted from checkerboard, Lighthouse (with two base stations), and MoiréBoard are plotted in red, green, and blue, respectively, while the camera undergoes a linear motion. The cyan box shows a zoom-in view of the visualization. **(a)** Comparison in the t_z range of 1.0m to 1.5m. **(b)** Comparison in the t_z range of 2.5m to 3.0m. It is evident that the tracking results of MoiréBoard always stay closely to the state-of-the-art Lighthouse 2.0 system.

When using two base stations, we place the stations symmetrically: they are centered at the same location as the MoiréBoard and separated by 1 meter. To understand the tracking accuracy at different distances from the landmark (i.e., MoiréBoard, checkerboard, or base stations), we divide the space into four regions (see Fig. 10). In each region, we place the linear slide along a random direction and evaluate the tracking errors of different methods (as described above); we repeat this process 10 times, each with a different slide direction.

The average tracking errors are reported in Table 4, where "MoiréBoard-1" and "MoiréBoard-6" indicate the use of MoiréBoard designs in the 1st and 6th row of Table 1, respectively. The results confirm that the tracking accuracy of MoiréBoard is an order of magnitude higher than the classic checkerboard-based approach. Our method is also much more accurate than Lighthouse 2.0 with a single base station. When Lighthouse 2.0 uses two base stations, both methods have comparable accuracy, although Lighthouse 2.0 requires power input to actively emit laser, and is much more costly. We also note that a single MoiréBoard has a more limited tracking range than Lighthouse 2.0 due to the reason discussed in Fig. 7. However, its tracking range can be easily extended by using two or more MoiréBoard designs (e.g., MoiréBoard-1 and MoiréBoard-6) in the same scene.

In terms of the computational cost, MoiréBoard only requires simple image processing operations before applying the tracking formula (6). Thereby, the algorithm is computationally lightweight, able to track at a much higher frame rate than the checkerboard approach (which needs to solve a PnP problem at every frame). The tracking frame rates are also reported in Table 4. We note that the image processing operations in our algorithm can be further accelerated using GPUs on mobile devices.

We also visualize the tracking errors of different methods in Fig. 11. From the plots, it is evident that **(i)** MoiréBoard tracking results match closely to the Lighthouse 2.0 with two base stations, and **(ii)** as the distance from the landmark increases, checkerboard-based tracking deteriorates quickly, while MoiréBoard stays closely with Lighthouse 2.0.

VR application. In addition, we build a mobile VR application using an iPhone 11 Pro placed on a Head Mounted Device (HMD). We use the mobile phone's camera to capture MoiréBoard in realtime and recover the HMD's position. Together with the HMD's orientation obtained from the phone's IMU sensor, our VR application offers a full 6-DoF VR experience. We refer the reader to the supplementary video for more details.

	Distance Range (m)				
Tracking Method	1.0-1.5	1.5-2.0	2.0-2.5	2.5-3.0	FPS
CheckerBoard	8.24	12.62	17.89	23.71	15.7
Lighthouse (1 Base)	5.62	7.98	9.03	13.31	60
Lighthouse (2 Base)	1.79	1.84	2.32	2.56	60
MoiréBoard-1	1.54	2.05	-	-	114.2
MoiréBoard-6	-	2.34	2.52	3.17	114.2

Table 3. **Real-scene tracking results.** We report the average tracking errors (in mm) of different methods as well as their tracking frame rate (in FPS). The checkerboard and MoiréBoard algorithms are implemented using iOS-OpenCV on an iPhone 11 Pro Max. Lighthouse algorithm is encapsulated in the hardware of Valve Index VR kit (Thus, its FPS may not reflect its true computational cost). Also note that our method can be further accelerated using the mobile device's GPUs, since the performance bottleneck of our algorithm lays in the image processing tasks (such as applying the Gaussian filter), which are well suited for parallel processing.

	Distance Range (m)				
Tracking Method	1.0-1.5	1.5-2.0	2.0-2.5	2.5-3.0	FPS
CheckerBoard	8.24	12.62	17.89	23.71	15.7
Lighthouse (1 Base)	5.62	7.98	9.03	13.31	60
Lighthouse (2 Base)	1.79	1.84	2.32	2.56	60
MoiréBoard	1.54	2.05	2.52	3.17	114.2

Table 4. **Real-scene tracking results.** We report the average tracking errors (in mm) of different methods as well as their tracking frame rate (in FPS). The checkerboard and MoiréBoard algorithms are implemented using iOS-OpenCV on an iPhone 11 Pro Max. Lighthouse algorithm is encapsulated in the hardware of Valve Index VR kit (Thus, its FPS may not reflect its true computational cost). Also note that our method can be further accelerated using the mobile device's GPUs, since the performance bottleneck of our algorithm lays in the image processing tasks (such as applying the Gaussian filter), which are well suited for parallel processing.

5 DISCUSSION AND CONCLUSION

We present a new camera tracking method that ultilizes moiré effects. Our method has high tracking accuracy, comparable to the state-of-the-art commercial VR tracking device. But our method is fully passive, requiring only two layers of repetitive structures, which can be made at low cost (such as printing on paper and 3D-printed meshes). It is therefore particularly suitable for low-cost VR systems such as Google Cardboard.

As a marker-based tracking method, MoiréBoard also shares some limitations with other marker-based approaches. When the camera moves at a high speed, the captured images may suffer from motion blur, which causes the image-plane measurement to become less robust. One solution to this issue is to use high frame rate to capture images and thereby reduce motion blur. Currently, most mobile devices can record video with a high FPS (usually up to 120FPS, with high-end devices such as the iPhone 11 can record at 240FPS). When the camera captures at 120FPS, our experiments show that the motion blur won't affect tracking accuracy if the moving speed is below 30cm/s, which is indeed the case for head motions in VR applications. Also, with sufficient lighting, the camera's shutter can be much faster than 1/120s regardless of its capture frame rate, further reducing the motion blur.

Another limitation of marker-based tracking stems from the camera's limited field-of-view (FoV). The optical landmark may not always be visible on the captured image. One possible solution is to use a wide angle camera (e.g., a fisheye camera and 360° camera). But this demands a more sophisticated image processing algorithm to rectify images due to wide-angle lens distortion.

Also related to the image rectification is the need of four visual markers at the corners of MoiréBoard. Currently, we rely on the markers to convert captured moiré fringes into a frontal view. However, we know *a priori* that all the moiré peak bands in the frontal view form square grids. Based on this prior, it is possible to rectify the images directly from

the moiré fringes, without the need of visual markers. Thereby, we can reduce the form factor of the MoiréBoard. But this will require additional attention to identify the region where MoiréBoard exists.

Lastly, different MoiréBoard designs will have different tracking ranges (recall Table 1). We can use multiple MoiréBoard designs at the same time to extend the tracking range. Multiple boards also will easily help extend the FoV and improve the robustness under occlusions. In addition, it is possible to use multiple MoiréBoard to recover the camera's orientation based on how the captured moiré patterns are distorted under a rotated camera (recall Fig. 6). This requires future research on how to measure the moiré grid's distortion on the image. Better yet we could use an LED display (such as an iPad) to *dynamically* change the grating pattern (such as T_a) on grid A, and thereby choose the most suitable MoiréBoard designs based on the camera's current position.

ACKNOWLEDGMENTS

We thank the anonymous reviewers for their constructive feedback, as well as Zixiaofan Yang and Henrique Maia for their help on proofreading. This work was supported by the National Science Foundation (NSF-1910839).

REFERENCES

[1] Karan Ahuja, Sujeath Pareddy, Robert Xiao, Mayank Goel, and Chris Harrison. 2019. Lightanchors: Appropriating point lights for spatially-anchored augmented reality interfaces. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology*. 189–196.

[2] Alex M Andrew. 2001. Multiple view geometry in computer vision. *Kybernetes* (2001).

[3] Brian Armstrong, Thomas Verron, Lee Heppe, Jim Reynolds, and Karl Schmidt. 2002. RGR-3D: simple, cheap detection of 6-DOF pose for teleoperation, and robot programming and calibration. In *Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No. 02CH37292)*, Vol. 3. IEEE, 2938–2943.

[4] Samuel Banks, Richard Green, and Jun Junghyun. 2019. Use of Moiré Patterns in Camera Position Estimation. In *2019 International Conference on Image and Vision Computing New Zealand (IVCNZ)*. IEEE, 1–7.

[5] Gunilla Borgefors, Heinz-Otto Peitgen, John K Tsotsos, Ales Leonardis, Reinhard Klette, Katsushi Ikeuchi, Isaac Amidror, Thomas S Huang, Rachid Deriche, and Tianzi Jiang. 2009. *The Theory of the Moiré Phenomenon*. Springer London.

[6] Carlos Campos, Richard Elvira, Juan J Gómez Rodríguez, José MM Montiel, and Juan D Tardós. 2020. ORB-SLAM3: An accurate open-source library for visual, visual-inertial and multi-map SLAM. *arXiv preprint arXiv:2007.11898* (2020).

[7] Andrew I Comport, Eric Marchand, Muriel Pressigout, and Francois Chaumette. 2006. Real-time markerless tracking for augmented reality: the virtual visual servoing framework. *IEEE Transactions on visualization and computer graphics* 12, 4 (2006), 615–628.

[8] HTC Corporation. 2017. HTC Vive VR System. https://www.vive.com/us/.

[9] Valve Corporation. 2019. Valve Index Lighthouse Tracking System. https://www.valvesoftware.com/en/index/base-stations.

[10] Joseph DeGol, Timothy Bretl, and Derek Hoiem. 2017. Chromatag: A colored marker and fast detection algorithm. In *Proceedings of the IEEE International Conference on Computer Vision*. 1472–1481.

[11] Paul Dempsey. 2016. The teardown: HTC Vive VR headset. *Engineering & Technology* 11, 7-8 (2016), 80–81.

[12] Ramon Estana and Heinz Woern. 2003. Moire-based positioning system for microrobots. In *Optical Measurement Systems for Industrial Inspection III*, Vol. 5144. International Society for Optics and Photonics, 431–442.

[13] LLC. Facebook Technologies. 2020. Oculus Quest 2. https://www.oculus.com/quest-2/.

[14] Mark Fiala. 2005. ARTag, a fiducial marker system using digital techniques. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, Vol. 2. IEEE, 590–596.

[15] Blender Foundation. 2021. Blender Software. https://www.blender.org/.

[16] Sergio Garrido-Jurado, Rafael Muñoz-Salinas, Francisco José Madrid-Cuevas, and Manuel Jesús Marín-Jiménez. 2014. Automatic generation and detection of highly reliable fiducial markers under occlusion. *Pattern Recognition* 47, 6 (2014), 2280–2292.

[17] Google. 2016. Google Cardboard Portable VR System. https://arvr.google.com/cardboard/.

[18] Elsayed E Hemayed. 2003. A survey of camera self-calibration. In *Proceedings of the IEEE Conference on Advanced Video and Signal Based Surveillance, 2003.* IEEE, 351–357.

[19] Seongkook Heo, Jaehyun Han, Sangwon Choi, Seunghwan Lee, Geehyuk Lee, Hyong-Euk Lee, SangHyun Kim, Won-Chul Bang, DoKyoon Kim, and ChangYeong Kim. 2011. IrCube tracker: an optical 6-DOF tracker based on LED directivity. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*. 577–586.

[20] Roger David Hersch and Sylvain Chosson. 2004. Band moiré images. *ACM Transactions on Graphics (TOG)* 23, 3 (2004), 239–247.

[21] Patrick Hübner, Kate Clintworth, Qingyi Liu, Martin Weinmann, and Sven Wursthorn. 2020. Evaluation of HoloLens tracking and depth sensing for indoor mapping applications. *Sensors* 20, 4 (2020), 1021.

[22] Michail Kalaitzakis, Sabrina Carroll, Anand Ambrosi, Camden Whitehead, and Nikolaos Vitzilaios. 2020. Experimental Comparison of Fiducial Markers for Pose Estimation. In *2020 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 781–789.

[23] Georg Klein and David Murray. 2009. Parallel tracking and mapping on a camera phone. In *2009 8th IEEE International Symposium on Mixed and Augmented Reality*. IEEE, 83–86.

[24] Manon Kok, Jeroen D Hol, and Thomas B Schön. 2017. Using inertial sensors for position and orientation estimation. *arXiv preprint arXiv:1704.06053* (2017).

[25] Steven M LaValle, Anna Yershova, Max Katsev, and Michael Antonov. 2014. Head tracking for the Oculus Rift. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 187–194.

[26] Vincent Lepetit, Francesc Moreno-Noguer, and Pascal Fua. 2009. Epnp: An accurate o (n) solution to the pnp problem. *International journal of computer vision* 81, 2 (2009), 155.

[27] Sony Interactive Entertainment LLC. 2016. Sony PlayStation VR. https://www.playstation.com/en-us/ps-vr/.

[28] Inc. Magic Leap. 2018. Magic Leap 1. https://www.magicleap.com/en-us/magic-leap-1.

[29] Wenguang Mao, Jian He, and Lili Qiu. 2016. Cat: high-precision acoustic motion tracking. In *Proceedings of the 22nd Annual International Conference on Mobile Computing and Networking*. 69–81.

[30] Microsoft. 2018. Hololens. https://www.microsoft.com/en-us/hololens/hardware.

[31] Raul Mur-Artal and Juan D Tardós. 2017. Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. *IEEE Transactions on Robotics* 33, 5 (2017), 1255–1262.

[32] Diederick C Niehorster, Li Li, and Markus Lappe. 2017. The accuracy and precision of position and orientation tracking in the HTC vive virtual reality system for scientific research. *i-Perception* 8, 3 (2017), 2041669517708205.

[33] Edwin Olson. 2011. AprilTag: A robust and flexible visual fiducial system. In *2011 IEEE International Conference on Robotics and Automation*. IEEE, 3400–3407.

[34] Daniel Eger Passos and Bernhard Jung. 2020. Measuring the accuracy of inside-out tracking in XR devices using a high-precision robotic arm. In *International Conference on Human-Computer Interaction*. Springer, 19–26.

[35] Daniel Pustka, Jan-Patrick Hülß, Jochen Willneff, Frieder Pankratz, Manuel Huber, and Gudrun Klinker. 2012. Optical outside-in tracking using unmodified mobile phones. In *2012 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. IEEE, 81–89.

[36] Samsung. 2017. Samsung Gear VR. https://www.samsung.com/global/galaxy/gear-vr/.

[37] Muhamad Risqi U Saputra, Andrew Markham, and Niki Trigoni. 2018. Visual SLAM and structure from motion in dynamic environments: A survey. *ACM Computing Surveys (CSUR)* 51, 2 (2018), 1–36.

[38] Takafumi Taketomi, Hideaki Uchiyama, and Sei Ikeda. 2017. Visual SLAM algorithms: a survey from 2010 to 2016. *IPSJ Transactions on Computer Vision and Applications* 9, 1 (2017), 1–11.

[39] Hideyuki Tanaka, Yasushi Sumi, and Yoshio Matsumoto. 2012. A high-accuracy visual marker based on a microlens array. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 4192–4197.

[40] Hideyuki Tanaka, Yasushi Sumi, and Yoshio Matsumoto. 2012. A visual marker for precise pose estimation based on lenticular lenses. In *2012 IEEE International Conference on Robotics and Automation*. IEEE, 5222–5227.

[41] Anran Wang and Shyamnath Gollakota. 2019. Millisonic: Pushing the limits of acoustic motion tracking. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. 1–11.

[42] Joshua T Weinhandl, Brian SR Armstrong, Todd P Kusik, Robb T Barrows, and Kristian M O'Connor. 2010. Validation of a single camera three-dimensional motion tracking system. *Journal of biomechanics* 43, 7 (2010), 1437–1440.

[43] Eric Whitmire, Farshid Salemi Parizi, and Shwetak Patel. 2019. Aura: Inside-out electromagnetic controller tracking. In *Proceedings of the 17th Annual International Conference on Mobile Systems, Applications, and Services*. 300–312.

[44] Robert Xiao, Chris Harrison, Karl DD Willis, Ivan Poupyrev, and Scott E Hudson. 2013. Lumitrack: low cost, high precision, high speed tracking with projected m-sequences. In *Proceedings of the 26th annual ACM symposium on User interface software and technology*. 3–12.

[45] Georges Younes, Daniel Asmar, Elie Shammas, and John Zelek. 2017. Keyframe-based monocular SLAM: design, survey, and future directions. *Robotics and Autonomous Systems* 98 (2017), 67–88.

[46] Cheng Zhang, Qiuyue Xue, Anandghan Waghmare, Sumeet Jain, Yiming Pu, Sinan Hersek, Kent Lyons, Kenneth A Cunefare, Omer T Inan, and Gregory D Abowd. 2017. Soundtrak: Continuous 3d tracking of a finger using active acoustics. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 1, 2 (2017), 1–25.

[47] Zhengyou Zhang. 2000. A flexible new technique for camera calibration. *IEEE Transactions on pattern analysis and machine intelligence* 22, 11 (2000), 1330–1334.

A ERROR ANALYSIS OF t_z RECOVERY

To understand the accuracy for tracking in z-direction, we analyze the sensitivity of recovered t_z in Eq. (6) with respect to the image space measurement α (recall $\alpha = \tilde{T}_\mathrm{m}/\tilde{L}$). A small measurement error $\delta\alpha$ on α will cause an error of recovered t_z to be $t_z(\alpha+\delta\alpha) - t_z(\alpha) \approx t_z'(\alpha)\delta\alpha$, where $t_z'(\alpha)$ is the derivative of t_z expression in (6) with respect to α:

$$t_z'(\alpha) = \frac{LT_\mathrm{a}^2 T_\mathrm{b} h}{\left[T_\mathrm{a}T_\mathrm{b} + \alpha L(T_\mathrm{a} - T_\mathrm{b})\right]^2}. \tag{8}$$

Concretely, consider a MoiréBoard with $L = 300$mm, $T_\mathrm{a} = 2.6$mm, $T_\mathrm{b} = 2.5$mm, and $h = 100$mm. Provided $|\alpha| \in [0.1, 0.5]$, we can compute a theoretical upper bound of t_z': $t_z' < 5617$. Our measurement error of $\tilde{T}_\mathrm{m}$ is about 1 pixel, leading to $\delta\alpha \approx 0.1\%$ (recall that we scale the image so that $\tilde{L} = 1000$px). As a result, the theoretical upper bound of the t_z error is 5.61mm. Note that our experiments show that in practice the error is much smaller than this upper bound.

B DERIVATION OF EQ. (7)

Equation (7) can be derived in the following way:

$$\begin{aligned}
\eta &= |\Delta\tilde{x}_\mathrm{m}|\frac{t_z}{t_x f} \\
&= \frac{t_z T_\mathrm{a}}{|T_\mathrm{a}h - T_\mathrm{a}t_z + T_\mathrm{b}t_z|} \quad \text{using (5)} \\
&= |\tilde{T}_\mathrm{m}|\frac{t_z}{T_\mathrm{b} f} \quad \text{using (4)}
\end{aligned} \tag{9}$$

Recall α is defined as

$$\alpha = \frac{\tilde{T}_\mathrm{m}}{\tilde{L}} = \tilde{T}_\mathrm{m}\frac{t_z}{Lf} \quad \Rightarrow \quad |\tilde{T}_\mathrm{m}|\frac{t_z}{f} = |\alpha|L. \tag{10}$$

Substituting this result in (9), we arrive to Eq. (7).

C PROOF OF MOIRÉ FRINGES INVARIANT TO CAMERA ROTATION

Consider two grid pattern, grid A and grid B, and a camera C is located at $C = (x_c, y_c, z_c)$. Let P_a and P_b be points located at grid A and grid B, respectively. The projected points $\tilde{P_a}$ and $\tilde{P_b}$ is defined by $\tilde{P_a} = \mathbf{KR}[\mathbf{I}|-C]P_a$ and $\tilde{P_b} = \mathbf{KR}[\mathbf{I}|-C]P_b$. Now let us assume $\tilde{P_a} = \tilde{P_b}$, namely P_a and P_b are located in the same point in the image space, which means $\tilde{P_a}$ and $\tilde{P_a}$ is located on the moiré peaks. Now we only need to prove that when $\mathbf{R}$ changes to a new value $\mathbf{R}'$, $\tilde{P_a} = \tilde{P_b}$ is still valid. In this case, the moiré pattern will retain its appearance.

Assume $\mathbf{R} = \mathbf{I}$, if $P_a = (x_a, y_a, z_a)$, we have $\tilde{P_a} = (\frac{f_x(x_a - x_c)}{z_a - z_c} + c_x, \frac{f_y(y_a - y_c)}{z_a - z_c} + c_y)$. Similarly, if $P_b = (x_b, y_b, z_b)$, we have $\tilde{P_b} = (\frac{f_x(x_b - x_c)}{z_b - z_c} + c_x, \frac{f_y(y_b - y_c)}{z_b - z_c} + c_y)$. Since $\tilde{P_a} = \tilde{P_b}$, we can easily obtain following relation:

$$\frac{x_a - x_c}{x_b - x_c} = \frac{y_a - y_c}{y_b - y_c} = \frac{z_a - z_c}{z_b - z_c}. \tag{11}$$

Now assume the camera has rotated θ around the $x-$axis so that $\mathbf{R} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{bmatrix}$.

In this case, we can derive that the x coordinate of $\tilde{P_a}$ is $\frac{f_x(x_a-x_c)}{(y_a-y_c)\sin\theta+(z_a-z_c)\cos\theta}+c_x$. The x coordinate of $\tilde{P_b}$ can be derived in the same way by replacing x_a with x_b and y_a with y_b. Since $\tilde{P_a}=\tilde{P_b}$, we have

$$\frac{x_a-x_c}{(y_a-y_c)\sin\theta+(z_a-z_c)\cos\theta}=\frac{x_b-x_c}{(y_b-y_c)\sin\theta+(z_b-z_c)\cos\theta},$$

which is equivalent to

$$((x_a-x_c)(y_b-y_c)-(x_b-x_c)(y_a-y_c))\sin\theta=((x_b-x_c)(z_a-z_c)-(x_a-x_c)(z_b-z_c))\cos\theta. \tag{12}$$

According to Eq. (11), we know $(x_a-x_c)(y_b-y_c)=(x_b-x_c)(y_a-y_c)$ and $(x_b-x_c)(z_a-z_c)-(x_a-x_c)(z_b-z_c)$, so Eq. (12) is always valid no matter the value of θ. The y coordinate of $\tilde{P_a}$ and $\tilde{P_b}$ also has the same property.

Similarly, we can prove that camera rotation along y and z axis will also not break the equality $\tilde{P_a}=\tilde{P_b}$, this indicates camera rotation will not change the appearance of moiré fringes.