ISSN 0361-7688, Programming and Computer Software, 2018, Vol. 44, No. 3, pp. 148–158. © Pleiades Publishing, Ltd., 2018.
Original Russian Text © V.A. Bobkov, A.P. Kudryashov, S.V. Mel'man, 2018, published in Programmirovanie, 2018, Vol. 44, No. 3.

On the Recovery of Motion of Dynamic Objects from Stereo Images

V. A. Bobkov[a],*, A. P. Kudryashov[a],, and S. V. Mel'man[a],*****

[a] *Institute of Automation and Control Processes, Far East Branch of the Russian Academy of Sciences, Vladivostok, 690041 Russia*

**e-mail: bobkov@dvo.ru*

***e-mail: kudryashovA@dvo.ru*

****e-mail: melman@dvo.ru*

Received August 20, 2017

Abstract—An approach to the recovery of trajectories of objects in a dynamic scene from stereo images is proposed. The approach is based on the use of a point representation of objects, visual odometry, and a set of algorithms that produce point models of objects and calculate their trajectories using matched 3D point clouds. Results of numerical experiments for synthetic scenes are discussed.

Keywords: stereo images, dynamic objects, point model, identification of objects, trajectory, autonomous underwater vehicle, visual navigation, visualization

DOI: 10.1134/S0361768818030027

1. INTRODUCTION

The problem of the simultaneous recovery of the motion trajectory of an autonomous robot and the construction of a 3D model of the environment from video data has long been studied; it is known in the literature as Simultaneous Localization and Mapping (SLAM) or Structure from Motion (SfM). This problem is well studied for static scenes. However, in the case of dynamic scenes (with moving objects), the proposed methods are poorly suited or are not applicable at all, which narrows down the range of practical applications.

To solve this problem in the dynamic statement, various methods using silhouettes, color and stereo images, and illumination and motion extracted from video data to reconstruct the geometry of dynamic objects (DOs) in the scene were proposed. The majority of studies is based on the controlled environment in which the background is known and calibrated fixed cameras are used. Only a small number of approaches were proposed for more general statements (with certain limitations). For example, in [1] it is assumed that the scene includes only one dynamic object. In [2], the reconstruction quality is limited at the billboard level. In [3], this limitation is removed, but the method requires scene preprocessing aimed at obtaining structure information about the static part of the scene, which facilitates the further reconstruction of the dynamic scene.

A more general approach to the reconstruction and segmentation of complex dynamic scenes using a great number of moving cameras without the a priori knowledge about the scene was proposed in [4]. In this approach, dense depth maps for each camera are evaluated and then combined to reconstruct each dynamic object. A feature of this approach is the method of coarse initial segmentation and the dense reconstruction algorithm.

From the viewpoint of algorithmic simplicity, an efficient computationally inexpensive approach that makes it possible to use standard GPUs is based on the point representation of objects. For example, [5] uses a global point model for representing static and dynamic objects. Each point is characterized by a constantly updated confidence indicator, which can be used to evaluate that probability that the point belongs to a static or dynamic object. The dynamic objects are formed by combining points based on their texture and 3D geometric characteristics.

An effective approach to the reconstruction of complex dynamic scenes was proposed in [6]. A feature of this approach is the use of temporal coherence, multi-view segmentation, and geodesic star convexity constraint that provide robust joint segmentation and shape reconstruction of dynamic objects. In the approach to real-time reconstruction and visualization proposed in [7], the focus is on the use of the calculated surface curvature. It is claimed in [7] that this approach reduced the error in of recovering the cam-

era trajectory and improved the reconstruction quality.

Note that the existing solutions are insufficiently universal, and the software and algorithms are often insufficiently efficient for practical purposes.

In this paper, we propose an approach to solving the SfM from video data problem as applied to dynamic scenes. This approach is based on the use of visual odometry, point model of objects, and implementation of a set of novel algorithms designed for the identification and processing of static and dynamic objects of the scene without a priori knowledge about the static objects. The proposed approach is an elaboration of the results obtained by the authors of this paper for static scenes [8–11]. The focus is on the detection of dynamic objects in a point model of the scene, recovery of the camera trajectory, and calculation of trajectories of dynamic objects. The algorithmic implementation and estimate of its effectiveness are given for the specific case of autonomous underwater vehicles (AUVs).

2. STATEMENT OF THE PROBLEM

We assume that an autonomous underwater vehicle equipped with a stereo camera moves in an a priori unknown environment along an arbitrary trajectory and captures images at a prescribed frequency. The scene contains, in addition to the static part (unmoving objects), appearing and disappearing dynamic objects scene that move along a priori unknown trajectories. The geometric shapes of these objects are not known in advance, but we assume that their shapes do not change in the course of the motion; i.e., the dynamic objects are rigid. In the classical statement, this problem is known under the name of visual SLAM—simultaneous solution of two interrelated problems: accurate reconstruction of the robot (camera) motion and construction of a 3D model of the observed static environment from the captured video stream.

In the statement of this problem for dynamic scenes, it is required, in addition to the reconstruction of the robot (camera) trajectory and the static scene, calculate the trajectories of the moving objects and their 3D shapes. In this paper, we consider the first problem—reconstruction of trajectories of all dynamic objects and the AUV trajectory. The solution is based on the visual odometry method. The algorithms proposed in this paper are adapted for the conditions under which AUVs operate; however, the proposed approach is universal.

The results of solving this problem in the form of matrix transformations determining the motion of the objects will be used for the animated visualization of the robot motion in the reconstructed 3D environment along an arbitrary prescribed trajectory. Such a visualization is needed for solving practical problems related to the analysis of situations in 3D scenes and execution of autonomous vehicle missions.

3. DESCRIPTION OF THE APPROACH

To calculate the trajectories of the camera and all dynamic objects, the dynamic objects must be detected and identified in each position of the robot. An additional difficulty in the calculation of the proper motion of dynamic objects is that their motion detected in the images is the result of the simultaneous motion of the camera and the object's proper motion. To describe the dynamic objects and the corresponding algorithms, we will use a point description of the scene objects. The set of 3D points for the subsequent processing is constructed based on the feature points matched in stereo pairs using the SURF detector or the Kanade–Lucas–Tomasi feature tracker (KLT tracker). To estimate the motion and reconstruct the trajectories from the captured video stream, the visual odometry method is used. According to this method, the classical computation scheme for calculating the camera trajectory relative to the static part of the scene uses the following step-by-step processing the stereo pair images:

• Matching 2D features in the pairs of sequential images corresponding to two adjacent positions on the trajectory in which the coordinates will be calculated.

• Construction of the set of 3D points (3D cloud) given the matched set of features in the stereo pair captured in the current position. Similarly, the 3D cloud for the stereo pair captured in the preceding position is constructed. The coordinates of the points in each 3D cloud are specified in the local system of coordinates related to the current position.

• Given the matched 3D clouds, calculation of the geometric transformation matrix describing the relative displacement of the camera (robot) (6 degrees of freedom (6DOF)) in the local system of coordinates.

The key task in the considered statement of the problem is to partition the points of the initial set into the points belonging to the static part of the scene and to dynamic objects.

The approach is based on the two-phase processing of the point representation of the scene at each designed point in time. In the first phase, a coarse partitioning of the points into groups supposedly belonging to the static part of the scene and to dynamic objects is made. The main criterion of this partitioning is the rigidity of dynamic objects, which allows us to evaluate the similarity of motions. The algorithmic selection of the static part of the scene in this phase helps obtain a more accurate description of the dynamic groups in the second phase. A problem in its own right is matching the dynamic group formed at the current step with the dynamic group formed at the preceding step. This problem is solved by checking the membership of the tested point in the spatial hull of an

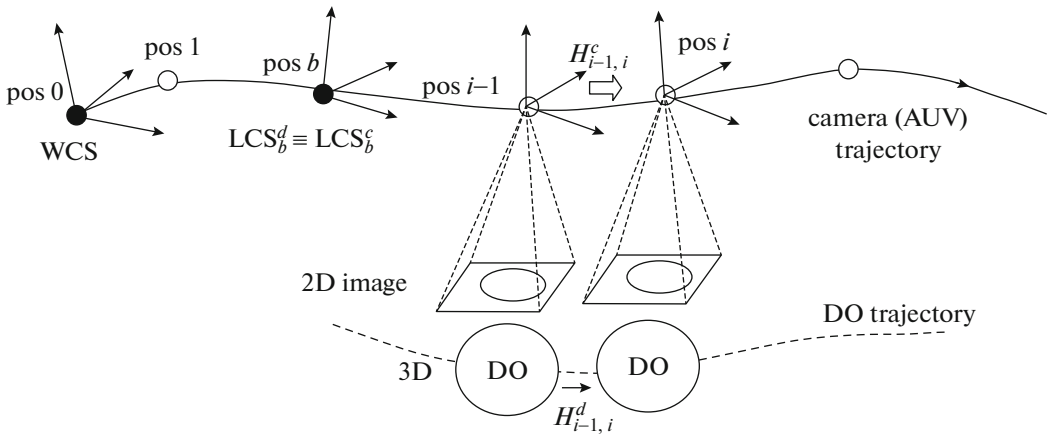

Fig. 1. The local coordinate systems (LCS^c) and the world coordinate system (WCS) used in the scene with dynamic objects (DO) when the camera (AUV) moves along a trajectory. Position b indicates the beginning of the DO motion in the scene.

earlier formed group and using the criterion of motion similarity.

Since the images captured by the camera reflect the result of the simultaneous motion of the camera and the objects, the computation of the local matrices determining the proper motion of each object is based on the use the computed matrix for the static part of the scene (the camera motion).

The set of algorithms implementing the proposed approach and the computation procedure are considered below.

4. ON THE RELATIONSHIP BETWEEN THE GEOMETRIC TRANSFORMATION MATRICES DETERMINING THE MOTION OF THE CAMERA AND THE MOTION OF DYNAMIC OBJECTS

The parameters (6DOF) of the AUV trajectory are computed in the positions i, where $i = 0,1,\ldots n$, and t_i is the time instant of the AUV motion corresponding to the position i of the AUV. Each position i is associated with the stereo pair of images captured by the camera from this position.

WCS is the world system of coordinates. Without loss of generality, we consider the local system of coordinates attached to the camera at the initial time as the *WCS*.

LCS_i^c is the local system of coordinates of the camera in position i, and d is the identifier of a dynamic object.

Each dynamic object d is characterized by its lifetime in the scene from the time t_b when it appears in the scene to the time t_e at which it leaves the scene. With the time t_b, we associate the initial system of coordinates LCS_b^d of the object d (Fig. 1).

In this system of coordinates, we can describe the proper motion of the object without reference to the camera motion. Furthermore, we will generate in this system of coordinates the full 3D model of this object; i.e., all the views of the object and their combination will be represented in this system. Without loss of generality, we can use the camera (AUV) local system of coordinates LCS_b^c in the position b as the initial system. Since the visual odometry method relates each LCS_i^c to the *WCS*, the coordinates of the dynamic object points can be recalculated to the *WCS*. For this reason, the proper relative (local) displacement of the dynamic object from $(i-1)$ to position i will be estimated by computing the geometric transformation matrix between the systems of coordinates LCS_{i-1}^c and LCS_i^c.

Note that the camera captures the scene image that is the result of simultaneous motion of the camera and the proper motions of the dynamic objects relative to the static part of the scene. To find the relation between these two motions, we use the fact that the motion of an object relative to the camera can be assigned a symmetric motion of the camera around the object. Then, the local complex transformation computed using the Iterative Closest Point (ICP) algorithm (based on the 3D clouds of adjacent positions constructed from the tracked object features) can be represented by

$$H_{i-1,i} = H_{i-1,i}^c H_{i-1,i}^d, \tag{1}$$

where $H_{i-1,i}$ is the complex transformation matrix that represents the result of the simultaneous motion of the camera and the dynamic object at the current step of the trajectory (this matrix is computed by the ICP algorithm),

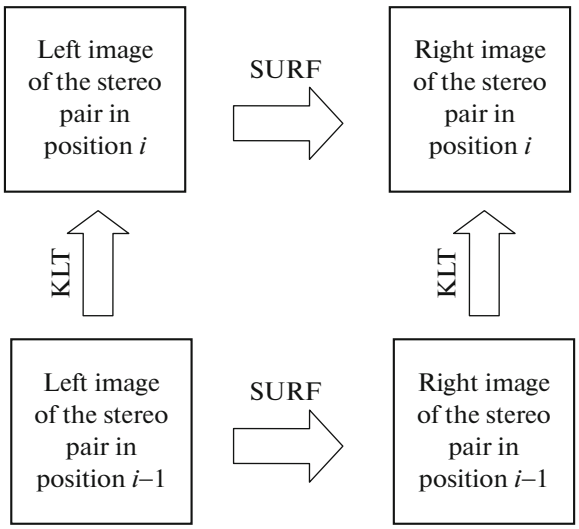

Fig. 2. Block diagram of processing images of two stereo pairs.

$H^c_{i-1,i}$ is the transformation matrix of the static point coordinates from the LCS^c_{i-1} system of coordinates to LCS^c_i describing the camera motion (it is computed from the identified static group),

$H^d_{i-1,i}$ is the transformation matrix of the dynamic object from the system of coordinates of position $(i-1)$ to the system of coordinates of position i describing the proper local displacement of the dynamic object.

Equation (1) implies that

$$H^d_{i-1,i} = H_{i-1,i}(H^c_{i-1,i})^{-1}. \quad (2)$$

That is, given the measurements (the result of the camera and dynamic object motion) and the computed motion of the camera, we can calculate the proper relative motion of the dynamic object. The relation between the 3D coordinates of each point P^d_{i-1} of the object d specified in the local system of coordinates of position $(i-1)$ and the corresponding point in position i is given by the equation $P^d_i = P^d_{i-1}H^d_{i-1,i}$.

The transformation matrix from the initial system of coordinates of the dynamic object to the system of coordinates of the current position i (the proper relative motion) is obtained by multiplying the corresponding local matrices:

$$H^d_{b,i} = H^d_{b,b+1}H^d_{b+1,b+2}\dots H^d_{i-1,i}. \quad (3)$$

Then, the coordinates of the object d at the time t_i are related to the coordinates of the initial system by the matrix $H^d_{b,i}$:

$$P^d_i = P^d_b H^d_{b,i}.$$

Hence, the inverse transformation $(H^d_{b,i})^{-1}$ makes it possible to place all views i of the object d into its initial system of coordinates.

To place all dynamic objects and their trajectories into the same system of coordinates (*WCS*), we should transform their coordinates from the initial system of coordinates of these objects to the *WCS*. Taking into account (3), this transformation can be written as

$$H^d_{i,WCS} = (H^c_{i,WCS})^{-1}(H^d_{b,i})^{-1}, \quad (4)$$

where $H^c_{i,WCS} = H^c_{0,1}H^c_{1,2}\dots H^c_{b-1,b}$, because we use the local system of coordinates of the camera at the initial time as the *WCS*. Thus, the ability to recalculate the coordinates of each dynamic object to the *WCS* allows us to reconstruct their trajectories and 3D models in the unified coordinate space of the world system of coordinates.

5. BASIC SET OF ALGORITHMS

5.1. Partition of the Initial Set of 3D Points Into Groups Belonging to Different Objects

For identifying the scene objects and tracking their motion in time, we will use the point representation of static and dynamic objects.

That is, at each point in time (at the position from which the stereo pair of images is captured by the camera), each object is represented by a set of spatial points belonging to the visible surfaces of the object. These points provide the original set both for solving the navigation problem and for reconstructing 3D objects. As has been mentioned above, the original set of 3D points is generated by the SURF detector and KLT tracker. Using the pairwise matching of the set of features in four images in two stereo pairs (Fig. 2), a unified matched set of features is generated; using this set, two 3D clouds are constructed using the ray triangulation method in positions $(i-1)$ and i. The set of object points in the current position can be represented by one or several groups of points using an algorithmic implementation of the proposed point selection procedure. The principles used to form these groups are as follows: (a) identification of the group by its initial point chosen from the original set of 3D points and belonging to an object in the scene (later the identification of this object is preserved by tracking it in time); (b) selection of points from the original set satisfying a criterion (criteria) that they belong to the given object. The initial point will be called the seed point.

The seed point is chosen by a special algorithm ensuring that it is located at a place with a high local density and that its matching error in two 3D clouds is minimal. For differentiating points of different objects

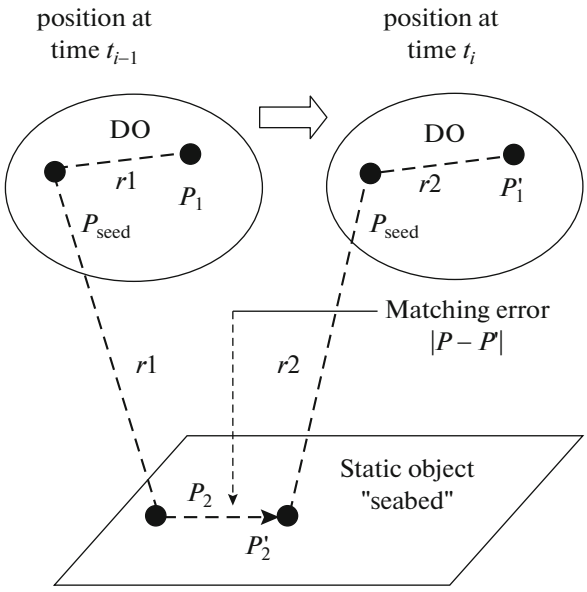

Fig. 3a. P_1 is the true DO point. P_2 is the false point in the DO group satisfying the rigidity criterion because $r1 = r2$.

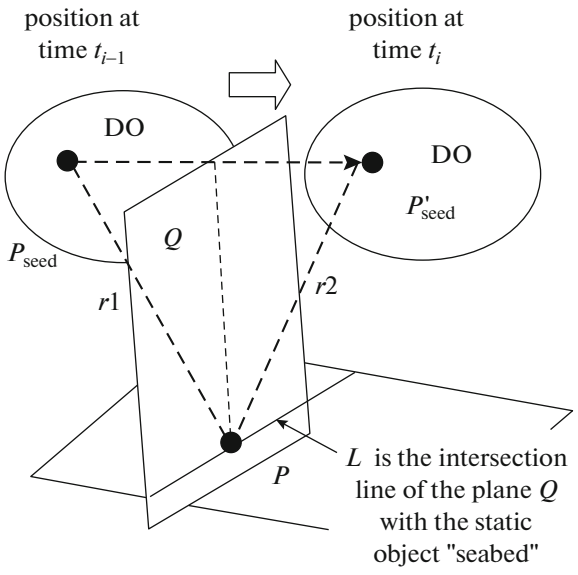

Fig. 3b. False points P occurring due to geometric ambiguity of the rigidity criterion lie on the spatial line L.

in the case when the scene objects are rigid, the natural criterion that the distances between the points of each object are preserved when it moves in the interval between two camera positions can be used. This criterion makes it possible to detect the motion of points of different objects. This criterion can be implemented in various ways. A simple and computationally efficient implementation is to compare the distances between the seed point and the point being tested in two 3D clouds. In this work, we deliberately did not use a threshold in the criterion implementation (because it has a degree of arbitrariness); instead, among all groups we selected the group that best satisfies the rigidity criterion; more precisely, we select the group with $\min|r1 - r2|$, where P_{seed} and P are the seed point of the group and the point being tested in the first 3D cloud, P'_{seed} and P' are, respectively, their matched images in the second 3D cloud, and

$$r1 = P_{seed} - P, \quad r2 = P'_{seed} - P'.$$

However, this criterion can produce false points. This can occur for two reasons: (a) matching error inherent in the detector used to match points in two clouds, which can be interpreted as an object motion; (b); the direction of motion of an object creates a geometric situation in which the distance between the seed point and the point being tested is the same in two 3D clouds. Two typical examples are illustrated in Figs. 3a and 3b. Figure 3a illustrates the geometric situation in which the point P_2 belonging to a static object is included in the group of dynamic objects due to a matching error made by the detector because it satisfies the rigidity criterion.

Figure 3b illustrates the situation of geometric ambiguity of the criterion when the points P on the spatial line L belonging to the static part of the scene satisfy the rigidity criterion applied to dynamic objects. The line L is the intersection of the plane Q and the seabed at which the camera is directed. The rigidity criterion is applied to the following pair of points: the seed P_{seed} (on the dynamic object) and the point P (on the seabed, which is in the static part of the scene). According to the distance criterion, if the point P belongs to the same object as P_{seed}, then $r1$ and $r2$ must be identical. However, the points P on the line L that belong to another object satisfy the criterion being verified because $r1 = r2$ due to the geometric construction. That is, for certain displacements of P_{seed} to P'_{seed}, ambiguous situations can occur in which the rigidity criterion does not unambiguously determine if two points belong to the same object. Such false points should be eliminated by filtering algorithms described below.

We have already mentioned that two partially different algorithms are used at the initial step and at the subsequent steps of trajectory calculation. In the first case, the points matched in two 3D clouds are partitioned into groups; in the second case, the static and dynamic groups identified in the current position are additionally matched to the corresponding groups identified in the preceding position. Both computation procedures are discussed below.

5.2. Algorithm for the Identification of the Group of Static Points

The idea underlying this algorithm is to classify a group of 3D points as belonging to the static part of the scene using a set of "weak" assumptions none of which gives a definite answer but makes a weighted contribution to the decision criterion.

The weak assumptions (criteria) are as follows:

• the maximal diagonal of the static group is equal to the diagonal of the scene enclosing box. The enclosing box is constructed by finding all min\ max coordinates over all points in the group of 2D and 3D points).

• the number of static points is much greater than the number of points in any dynamic object;

• in the case of underwater situations, the static (seabed) points are farther from the camera than the dynamic objects in the water mass.

The algorithm can be improved by adding additional criteria.

5.2.1. Partitioning the original set of points into static and dynamic. The matrix of the static part H_{static} calculated at the stage of the coarse partitioning of points into groups allows us to use the motion criterion for estimating the point status; this criterion partitions the original set of points into the static and dynamic groups more reliably. For a static point in the first 3D cloud, its image in the second 3D cloud can be calculated using the matrix H_{static}. If H_{static} is known exactly and there are no errors in matching features, then the calculated image must be identical to the matched point in the second cloud. If the matrix is known approximately, then differences in the matched points can occur, but they must not exceed a certain threshold. However, for dynamic points, this discrepancy is greater than the specified threshold. The criterion based on the estimation of the calculated discrepancy makes it possible to partition the original points into the static and dynamic subsets: if $\left|P \cdot H_{\text{static}} - P'\right| <$ *threshold*, then the point is static; otherwise, it is dynamic. Here P is the point in the first cloud, and P' is the corresponding point in the second cloud.

As a result, we obtain the refined static group and the other part D of the original set, which is the set of points belonging to dynamic objects. The resulting set of dynamic points is then filtered and partitioned into groups belonging to different dynamic objects. The algorithms used for this purpose are described below.

5.2.2. Algorithm for removing false points from the static group. Let

$C^{\text{surf}}_{\text{static}}$ be the set of the group of 3D static points in the current position;

$C^{\text{surf}}_{\text{dynamic}}$ be the set of the 3D dynamic points in the current position.

The validity of the membership of each point $P \in C^{\text{surf}}_{\text{static}}$ is checked. The point that actually belongs to the set of dynamic points $C^{\text{surf}}_{\text{dynamic}}$ is false. The validity criterion is the presence (absence) of dynamic points $\in C^{\text{surf}}_{\text{dynamic}}$ in a neighborhood of the point P being tested. That is, if there are dynamic points in a neighborhood of P, then the point P is decided to be false. To form the neighborhood without using a threshold (typically, the selection of a threshold has a certain degree of arbitrariness), we construct a 2D triangulation net using the projections of the static points onto the screen. Then, we can consider the set of triangles in the net with the vertices coinciding with P as its neighborhood. Then, checking the neighborhood for the existence of dynamic points is reduced to the regular application of the procedure that finds out if a given dynamic points belongs to a triangle.

5.2.3. Filtering algorithm for the set of dynamic points. Since the static group has already been defined at the stage of coarse partitioning of points into groups, we can use the points in this set to form a filtering criterion for the dynamic points. Effectively, the set of static points is the surface S seen by the camera to which the dynamic points should not adjoin. Therefore, for each tested point from the dynamic set, we can find out if it belongs to this surface or is near it to.

Determine the intersection point (P^S_{int}) of the ray outgoing the center of projections of the camera and passing through the point P_k with the surface S.

Apply the following criterion: if $|P_k - P^S_{\text{int}}| >$ threshold, then the point is dynamic; otherwise, it is false. The threshold characterizes the admissible error that can be made by the feature matching SURF detector and the procedure of constructing the corresponding 3D point.

5.2.4. Algorithm for forming dynamic groups in the current position and identifying them with the groups detected in the preceding position. To recover the trajectory of a dynamic object motion in the scene, we should match the point representations of dynamic objects that are independently formed at each position. Thus, this algorithm is designed to simultaneously accomplish two tasks—form dynamic groups at the current step and identify them with the dynamic groups found at the preceding step.

Let

M_i be the set of dynamic 3D points found at step $[i, i+1]$ in position i;

M_{i+1} be the set of dynamic 3D points found at step $[i, i+1]$ in position $i+1$;

$g^{k,i-1}_{i-1,i}$ be the dynamic group number k formed at step $[i-1, i]$ in position $i-1$;

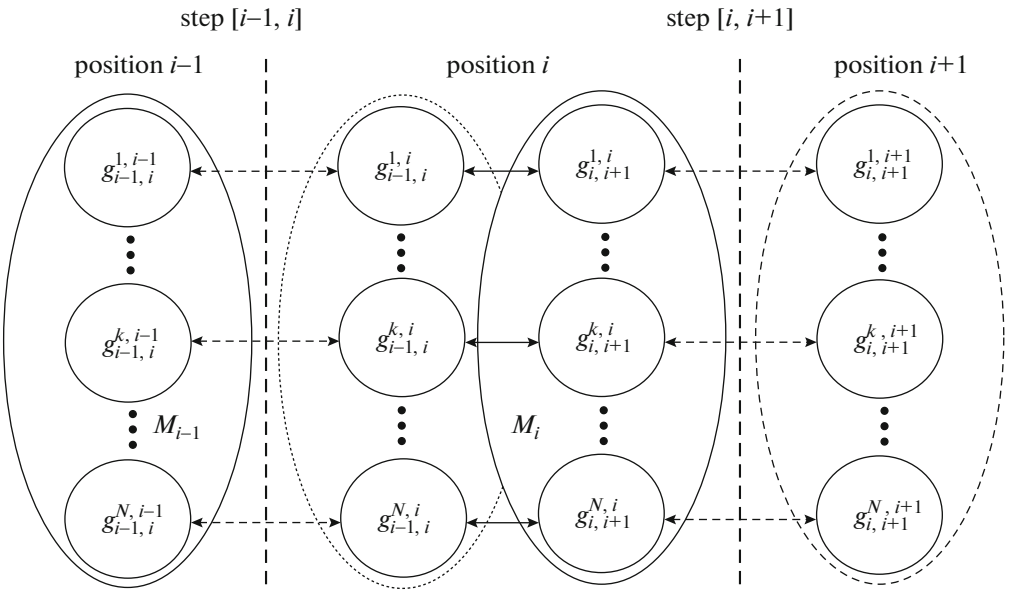

Fig. 4. Forming new dynamic groups (DGs) in position i at step $[i, i+1]$ and their association with the DGs in the preceding position.

$g^{k,i}_{i-1,i}$ be the group of points in position i assigned to the group $g^{k,i-1}_{i-1,i}$ in the 3D cloud;

$g^{k,i}_{i,i+1}$ be the dynamic group number k formed at step $[i, i+1]$ in position i;

$g^{k,i+1}_{i,i+1}$ be the group of points in position $i+1$ assigned to the group $g^{k,i}_{i,i+1}$ in the 3D cloud.

The initial data for the algorithm is the set of dynamic points M_i. Note that: (a) for each dynamic group $g^{k,i-1}_{i-1,i}$ in position $(i-1)$ formed at the preceding step $[i-1, i]$, there is its image $g^{k,i}_{i-1,i}$ (in the matching 3D cloud) in position i (see Fig. 4); (b) the points in the group $g^{k,i}_{i-1,i}$ and the points in the set M_i are specified in the same system of coordinates in position i. Therefore, for each tested point P in M_i, we can determine if it geometrically belongs to the group $g^{k,i}_{i-1,i}$, i.e., if it is inside the spatial hull enclosing the points of this group. If the point is inside this hull, then it belongs to the same dynamic object represented in position $(i-1)$ by the dynamic group with the index k. Thus, we establish the relationship of the point in the newly formed group in position i with an already existing dynamic group in the preceding position. Testing all the points in M_i with respect to each dynamic group detected in the preceding position makes it possible to partition the points M_i into dynamic groups and simultaneously identify them with the dynamic groups detected in the preceding position.

If no identification was made, then we conclude that a new dynamic object has appeared in the scene. In this case, the new dynamic group is registered in the list of groups. Thus, the list of groups reflects tracking of dynamic groups and the appearance of new and disappearance of earlier detected dynamic groups (i.e., objects).

The procedure just described applies if the spatial hulls mentioned above completely enclose the dynamic objects. However, it may happen that the hull encloses only a part of a dynamic object if the number of points in the corresponding dynamic group is not sufficiently large. In this case, some tested points that actually belong to this dynamic object can be outside the hull and will be treated as belonging to new objects. To take into account such situations, the procedure described above is modified as follows. The points in M_i that were not identified as belonging to a group detected at the preceding step using the *point in hull algorithm* are assigned to a special subset. For this subset, seed points are formed. Next, the points in this subset are processed using the algorithm described above using the rigidity criterion, which helps form new dynamic groups. The task is to identify the newly formed groups with the groups detected at the preceding step.

The proposed algorithm is based on computing the geometric correspondence measure between the just formed group and each group detected at the preced-

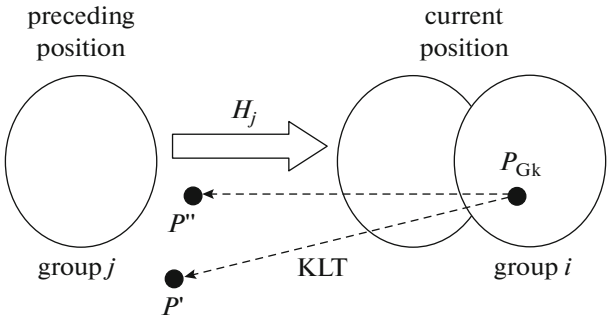

Fig. 5. Evaluation of mismatch with the group j in the preceding position for the point P_{Gk} belonging to the set G of the formed group i, $P'' = P_{Gk} \cdot H_j^{-1}$.

ing step. To compute this measure, the points at the preceding step group are first tracked using the KLT tracker. The result is a subset of corresponding points G. Then, the correspondence measure between group i at the current step and group j at the preceding step is computed as the sum of mismatches over all points in G. The mismatch for one point (see Fig. 5) is computed by the rule $\Delta_k^{ij} = |P_{Gk} \cdot H_j^{-1} - P'|$, where P_{Gk} is the kth point in G; H_j is the geometric transformation of the local system of coordinates in position $(i-1)$ to the local system of coordinates in position i, and P' is the point in position $(i-1)$ in the local system of coordinates assigned to the point P_{Gk} by the KLT tracker.

From the list of mismatches between the group i at the current step with the groups j at the preceding step, the minimal mismatch is chosen. If it does not exceed a predefined threshold, then we identify this group i with the group j with the minimal mismatch. Otherwise, the group corresponds to a new dynamic object.

The final step is the to merge the dynamic groups detected at the current step with respect to the similarity of their motion (comparison of their displacement vectors and the rotation quaternions in transformation matrices).

6. COMPUTATION PROCEDURE FOR PARTITIONING THE ORIGINAL SET OF 3D POINTS INTO GROUPS

6.1. Computation Procedure at the Initial Step of Trajectory Recovery

In contrast to the subsequent steps, the initial step of the trajectory processing does not require the detected groups of points to be identified with the groups detected at the preceding step.

The procedure (see Fig. 6) involves the following sequence of data processing steps using the algorithms described above:

1. Construction of the set of 3D points in the current position given the data provided by the SURF detector (matching points in four images taken in the current and the next positions of the trajectory).

2. Generation of seed points in the cells of the spatial grid of the camera field of view.

3. The initial (coarse) partitioning of the original set of 3D points into groups using the rigidity criterion (see Subsection 5.2.1).

4. Identification of the static group using the algorithm described in Section 5.2. Consolidation of groups. Computation of the camera motion matrix H_{static} from the points of the static group.

5. Partitioning the original set of points into two subsets of static and dynamic points using the algorithm described in Subsection 5.2.1. As a result, the refined static group and the set of dynamic points are obtained.

6. (a) Filtering the static group (Subsection 5.2.2) with the subsequent refinement of its local transformation matrix. (b) Filtering the static group (Subsection 5.2.3).

7. Partitioning the points of the set D into groups that supposedly belong to different dynamic objects: (a) generation of new seed points for the new dynamic set; (b) again apply the partitioning algorithm with the rigidity criterion to the dynamic set. After the dynamic groups have been formed, consolidate them based on the similarity of the computed geometric transformation matrices.

All the groups and dynamic objects found in the course of the trajectory processing are registered in a special table in which the lifetime of each dynamic object in the scene is specified. This data is needed for the computation of the transformation matrices between the local and world systems of coordinates and for solving the 3D reconstruction problem.

6.2. The Computation Procedure for Forming Dynamic Groups in the Current Position and Matching Them with the Groups Found in the Preceding Position

This computation procedure is executed at each step of the trajectory processing except for the first step, for which the initial step procedure is used (see the description above). This procedure involves the following sequence of data processing steps using the algorithms described above (see Fig. 6):

1. Execution of Steps 1–6 of the initial step computation procedure. As a result, the static group and the set of dynamic points are formed.

2. Formation of dynamic groups in position i using the algorithm of determining the membership of a point in a hull with the simultaneous assignment to the groups found at the preceding step (see Subsection 5.2.4). This procedure simultaneously forms the images of groups

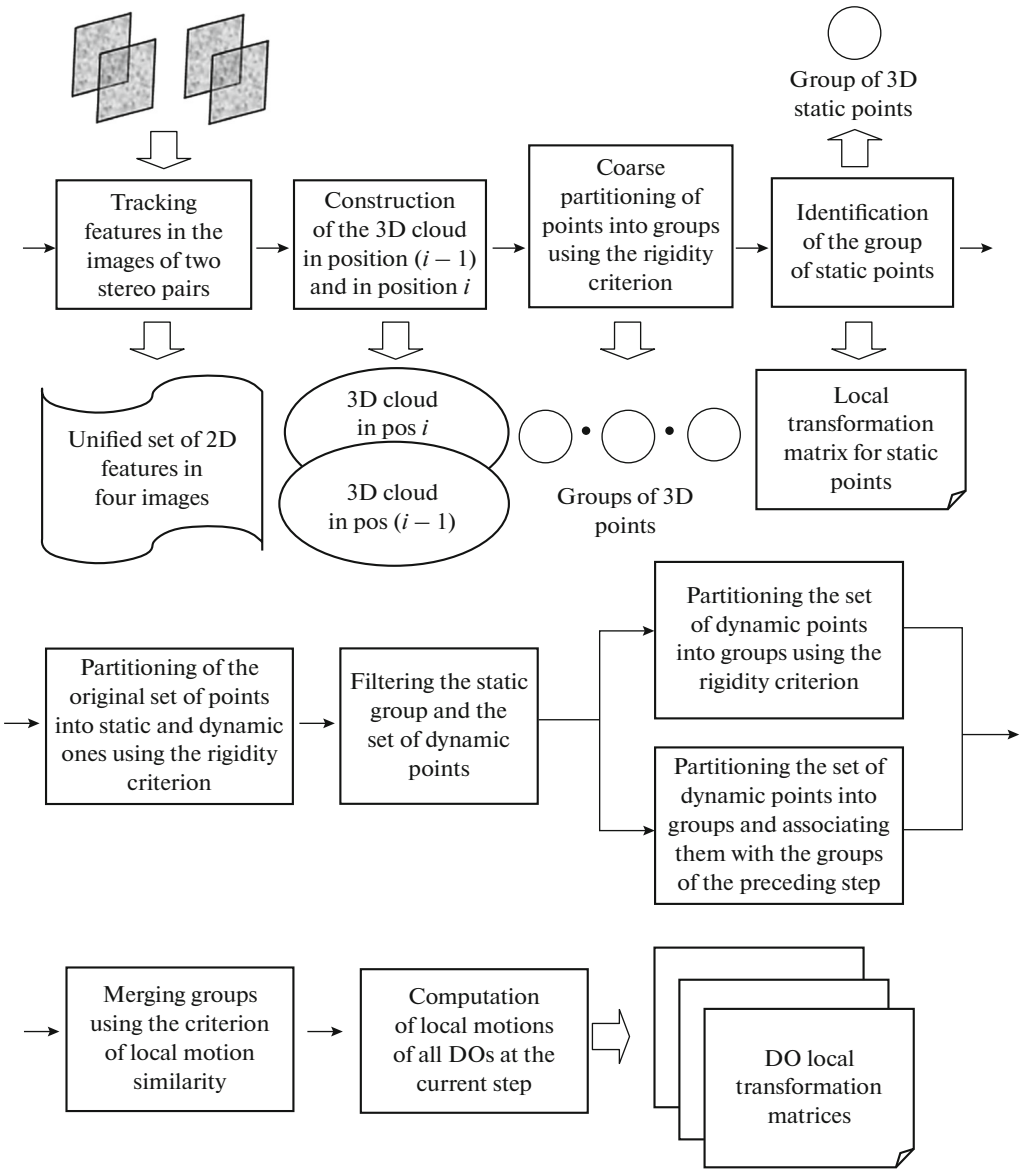

Fig. 6. Data processing procedure at one step of the trajectory.

also in position $(i + 1)$ based on the matching between the sets M_i and M_{i+1}. The groups obtained at the next step $[i + 1, i + 2]$ will be associated with the groups obtained at the current step.

7. EXPERIMENTS

Experiments on estimating the effectiveness of the proposed technology were performed using synthetic scenes (an example of such a scene is shown in Fig. 7).

Moving textured boxes over the modeled seabed topography were used as dynamic objects in the synthetic scenes. The speed of these objects was ≈ 0.5 m/s. The local error (the relative error for one step) and the absolute error (in the world system of coordinates) of the calculated trajectory of each dynamic object relative to the trajectory specified in the model were evaluated. The effectiveness of the algorithms at different stages of the computation procedure was evaluated. Figure 8 illustrates the result produced by the computation procedure at a step of partitioning the original set into groups corresponding to dynamic objects. The results of evaluating the accuracy of dynamic object localization are shown in Fig. 9. Error estimates for the camera localization were also obtained: the mean error of determining the local displacement was 0.01 cm, and the mean absolute error was 0.13 cm. It is seen from the plots that the error in

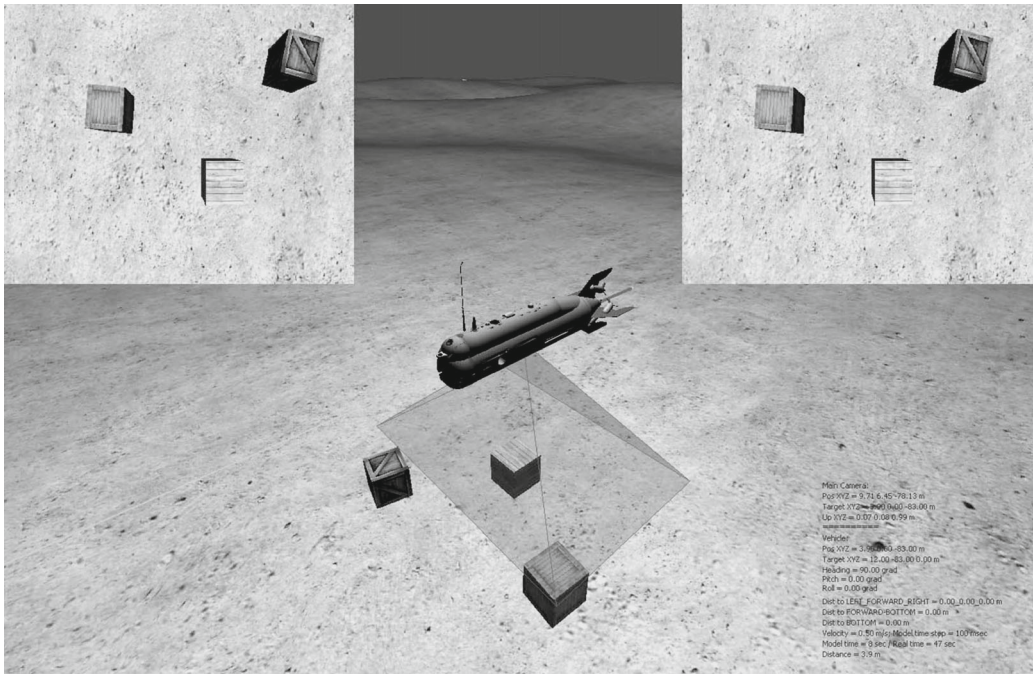

Fig. 7. Synthetic scene: capturing moving objects from the AUV trajectory.

the calculation of dynamic object trajectories is much higher than the error in the calculation of the camera trajectory because the number of points used to calculate the dynamic object motion is significantly lower than the number of points used for recovering the static part of the scene.

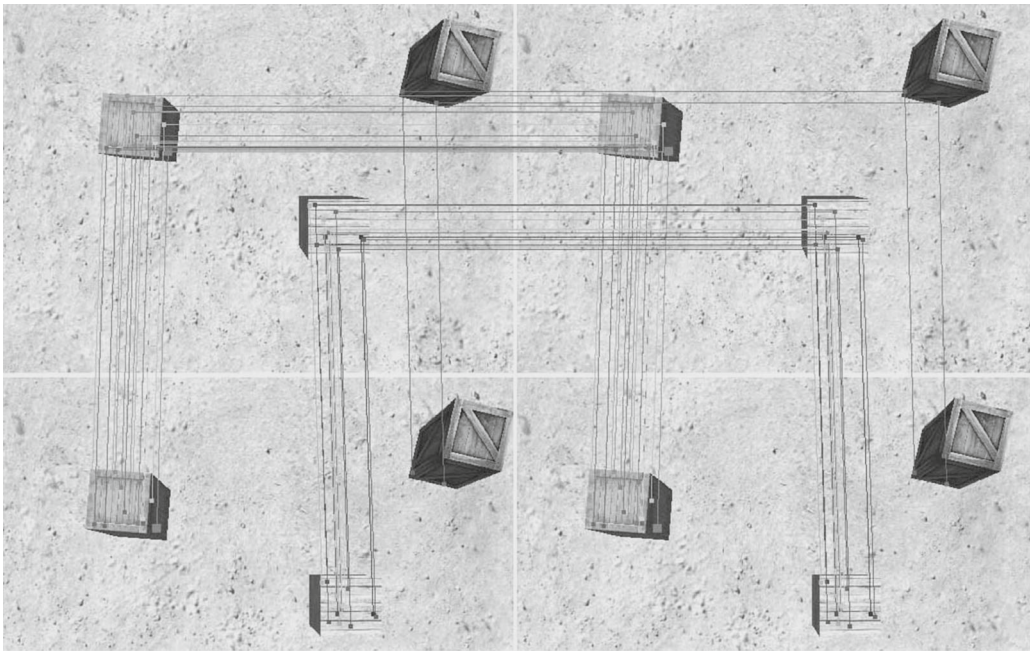

Fig. 8. Forming groups of points belonging to different DOs: The point features matched in four images of two stereo pairs are shown. Using these points, 3D points belonging to three different DOs are constructed. Correspondingly, three groups of points containing 147, 140, and 20 points are formed. In the figure, every tenth point is visualized to avoid overloading the figure with a large number of points.

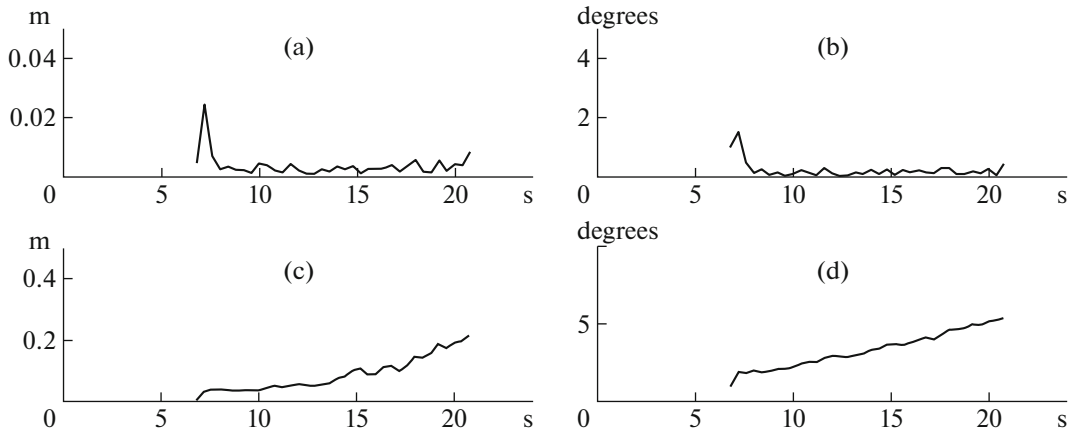

Fig. 9. Localization errors of a dynamic object: (a) the error in computing the local displacement; (b) the error in computing the local orientation change; (c) the error in computing the absolute displacement; (d) the error in computing the absolute orientation change.

8. CONCLUSIONS

The results of numerical experiments for synthetic scenes confirmed that the proposed approach is promising. However, the analysis of test results showed that the proposed computation procedure does not always generate a sufficiently large number of points in the models of dynamic objects. The insufficient number of points available for the computation of local matrices can increase the error in the localization of dynamic objects. In addition, for the high-quality 3D reconstruction of dynamic objects, a relatively high density of the point representation is required. For this reason, the further research can be directed at (a) the modification of the computation procedure to enable it to produce an extended point representation using the virtual range finder program developed by the authors and (b) the solution of the second part of problem, i.e., 3D reconstruction of dynamic objects using the computed trajectories objects when individual views of these objects are joined into a unified model.

ACKNOWLEDGMENTS

This work was supported by the Russian Foundation for basic research, project no. 18-07-00165.

REFERENCES

1. Hasler, N., Rosenhahn, B., Thormahlen, T., Wand, M., Gall, J., and Seidel. H.P.. Markerless motion capture with unsynchronized moving cameras, *Comput. Vision. Pattern Recognit.,* 2009, pp. 224–231.
2. Ballan, L., Brostow, G.J., Puwein, J., aand Pollefeys, M., Unstructured video-based rendering: Interactive exploration of casually captured videos, *ACM Trans. Graphics, Proc. of SIGGRAPH,* 2010, pp. 134–146.
3. Taneja, A., Ballan, L., and Pollefeys, M., Modeling dynamic scenes recorded with freely moving cameras, in *Conf. on Computer Vision,* 2011, pp. 613–626.
4. Mustafa, A., Kim, H., Guillemaut, J-Y., and Hilton, A., General dynamic scene reconstruction from multiple view video, in *Proc. of the IEEE Int. Conf. on Computer Vision,* 2015, pp. 900–908.
5. Keller, M., Lefloch, D., Lambers, M., Izadi, S., Weyrich, T., and Kolb, A., Real-time 3D reconstruction in dynamic scenes using point-based fusion, in *Proc. of the Joint 3DIM/3DPVT Conference (3DV),* 2013, pp. 1–8.
6. Mustafa, A., Kim, H., Guillemaut, J-Y., and Hilton, A., Temporally coherent 4D reconstruction of complex dynamic scenes, in *IEEE Conf. on Computer Vision and Pattern recognition,* 2016, pp. 223–245.
7. Lefloch, D., Kluge, M., Sarbolandi, H., Weyrich, T., and Kolb, A., Comprehensive use of curvature for robust and accurate online surface reconstruction, in *IEEE Trans. on Pattern Analysis and Machine Intelligence,* 2017.
8. Bobkov, V.A., Ron'shin, Yu.I., Kudryashov, A.P., and Mashentsev, V.Yu., 3D SLAM from Stereoimages, *Program. Comput. Software,* 2014, vol. **40**, no. 4, pp. 159–165.
9. Bobkov, V., Mashentsev, V., Tolstonogov, A., and Scherbatyuk, A., Adaptive method for AUV navigation using stereo vision, in *Proc. of the 26th ISOPE Int. Ocean and Polar Engineering Conference,* 2016.
10. Bobkov, V., Melman, S., Kudrashov, A., and Scherbatyuk, A., Vision-based navigation method for a local maneuvering of the autonomous underwater vehicle, in *IEEE OES Int. Symposium on Underwater Technology 2017 (UT 2017),* Busan, South Korea, pp.21–24.
11. Bobkov, V.A., Melman, S.V., and Kudryashov, A.P., Fast computation of local displacement by stereo pairs, *Pattern Recognit. Image Anal.,* 2017, no. 3, pp. 458–465.

Translated by A. Klimontovich