Efficient l_1-Norm-Based Low-Rank Matrix Approximations for Large-Scale Problems Using Alternating Rectified Gradient Method

Eunwoo Kim, *Student Member, IEEE*, Minsik Lee, *Member, IEEE*, Chong-Ho Choi, *Member, IEEE*, Nojun Kwak, *Member, IEEE*, and Songhwai Oh, *Member, IEEE*

***Abstract*— Low-rank matrix approximation plays an important role in the area of computer vision and image processing. Most of the conventional low-rank matrix approximation methods are based on the l_2-norm (Frobenius norm) with principal component analysis (PCA) being the most popular among them. However, this can give a poor approximation for data contaminated by outliers (including missing data), because the l_2-norm exaggerates the negative effect of outliers. Recently, to overcome this problem, various methods based on the l_1-norm, such as robust PCA methods, have been proposed for low-rank matrix approximation. Despite the robustness of the methods, they require heavy computational effort and substantial memory for high-dimensional data, which is impractical for real-world problems. In this paper, we propose two efficient low-rank factorization methods based on the l_1-norm that find proper projection and coefficient matrices using the alternating rectified gradient method. The proposed methods are applied to a number of low-rank matrix approximation problems to demonstrate their efficiency and robustness. The experimental results show that our proposals are efficient in both execution time and reconstruction performance unlike other state-of-the-art methods.**

***Index Terms*— Alternating rectified gradient method, l_1-norm, low-rank matrix approximation, matrix completion (MC), principal component analysis (PCA), proximal gradient method.**

I. Introduction

LOW-RANK matrix approximation has attracted much attention in the areas of subspace computation, data reconstruction, image denoising, and dimensionality reduction [1]–[10]. Since real-world data are usually high-dimensional, it is desirable to reduce the dimension of data without loss of information for rapid computation. Many data can usually be well represented with fewer parameters, and reducing the data dimension not only reduces the computing time, but also removes unwanted noise components. One of the most popular methods addressing these issues is principal component analysis (PCA) [11].

PCA transforms data to a low-dimensional subspace that maximizes the variance of given data based on the Euclidean distance (l_2-norm). The conventional l_2-norm-based PCA (l_2-PCA) that has been used in many problems is sensitive to outliers and missing data, because the l_2-norm can sometimes amplify the negative effects of such data. This prevents recognition or computer vision systems from performing well.

As an alternative, low-rank matrix approximation based on the l_1-norm, which is more robust to outliers [1]–[3], [5], [6], has been proposed. Also, there are many weighted low-rank matrix approximation methods based on the l_1-norm, in dealing with matrix completion (MC) problems, in the presence of missing data [3], [9], [10], [12]. These techniques assume a Laplacian noise model instead of a Gaussian noise model. In spite of the robustness of these approaches, they are too computationally intensive for a practical use and it is hard to find a good solution of the l_1 cost function because of its nonconvexity and nonsmoothness. There are alternative approaches [4], [7], [13]–[15] that modify the l_1 cost function to resolve the computation and robustness issue, but their solution are not as good as that obtained for the original l_1 cost function. Recently, robust PCA (RPCA) methods have been emerged to solve the non-Gaussian noise model and computation issues associated with large-scale problems based on various optimization methods [8], [12], [16]–[18] and these optimization methods extended to the tensor-based methods [19], [20].

Hawkins *et al*. [1] proposed a robust version of singular value decomposition (SVD) by applying an alternating l_1-regression algorithm and a weighted median technique to overcome outliers and missing data. Although this method is robust to such data, it takes a long time to find a local minimum using the alternating minimization technique. Ke and Kanade [2], [3] presented convex programming and weighted median approaches based on alternating minimization for the l_1-norm-based cost function, but these methods are also computationally expensive.

Ding *et al*. [4] proposed r_1-PCA, which combines the merits of the l_1-norm and l_2-norm. Kwak [5] suggested PCA-l_1 to

Manuscript received February 18, 2013; revised November 25, 2013 and March 11, 2014; accepted March 16, 2014. The work of E. Kim and S. Oh was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science, ICT and Future Planning, under Grant NRF-2013R1A1A2065551, and by the Seoul National University Research Grant. The work of N. Kwak was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science, ICT and Future Planning, under Grant NRF-2013R1A1A1006599. *(Corresponding author: M. Lee.)*

E. Kim, C.-H. Choi, and S. Oh are with the Department of Electrical and Computer Engineering, ASRI, Seoul National University, Seoul 151-744, Korea.

M. Lee and N. Kwak are with the Graduate School of Convergence Science and Technology, AICT, Seoul National University, Suwon 443-270, Korea (e-mail: mlee.paper@gmail.com).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNNLS.2014.2312535

find successive projections, using a greedy approach to maximize the l_1-norm of the projected data in the feature space. These two methods adopt modified cost functions, which are different from the original l_1 cost function given in [2] and [3], resulting in the degradation of robustness to outliers.

Brooks *et al*. [6] presented a method for finding successive, orthogonal l_1-norm-based hyperplanes to minimize the l_1-norm cost function. Eriksson and Hengel [9] proposed a weighted low-rank matrix approximation using the l_1-norm in the presence of missing data. However, these methods find the solution using convex linear programming, which is computation intensive and requires more memory storage.

Lin *et al*. [12] proposed RPCA based on the l_1-norm and nuclear norm for a nonfixed rank problem and solved it using the augmented Lagrange method (ALM). However, it requires too much execution time for large-scale problems because it performs SVD at each iteration. Toh *et al*. [16] proposed an accelerated proximal gradient method to find a robust solution for PCA problems. These methods find a solution for large-scale problems based on rank estimation, but they may not give a good solution for some computer vision problems such as structure from motion (SFM), which is a fixed-rank problem [10].

Shen *et al*. [8] proposed a low-rank matrix approximation using the l_1-norm based on augmented Lagrangian alternating direction method (ALADM). Zheng *et al*. [10] recently proposed a practical weighted low-rank approximation method using ALM. Both methods have fast convergence rates and can be applied to fixed-rank problems, but it requires significant parameter tuning for optimal performance.

In this paper, we propose two alternating rectified gradient algorithms that solve the l_1-based factorization problem at significantly less running time and memory for large-scale problems. Even though the proposed methods are based on an alternating minimization method, they give fast convergence rates owing to the novel method of finding the update direction by a rectified representation based on matrix orthogonalization. These methods are derived from the observation that there are numerous projections and coefficient matrices that give the same multiplication result while the convergence speed depends largely on how these matrices are chosen. After finding an update direction, we use the weighted median algorithm to find the step size for updating a matrix. Since the weighted median technique has small computation complexity, it is appropriate for large-scale problems. However, unlike the method in [2], which applies the weighted median algorithm columnwise, we apply it to the entire matrix at once to reduce the computational burden. The proposed methods are more efficient and robust than the other factorization and RPCA methods in solving various problems, as shown in Section V.

We demonstrate the competitiveness of the proposed methods in terms of the reconstruction error and computational speed, for examples, such as large-scale factorization problems and face reconstruction from occluded or missing images. In addition, the proposed methods are applied to a nonrigid SFM problem using two well-known multiview benchmark data sets.

This paper is organized as follows. In Section II, we briefly review an alternating-update approach of low-rank matrix approximation based on the l_1-norm and discuss the drawbacks of this approach. In Section III, we propose an approximated low-rank matrix approximation algorithm based on the l_1-norm. Then, we propose an improved algorithm based on dual formulation in Section IV. In Section V, we present various experimental results to evaluate the proposed method with respect to other well-known subspace analysis methods.

II. Preliminaries

In this section, we briefly review low-rank matrix approximation methods based on the l_1-norm.

Principal component analysis (PCA), which can be used to approximate a matrix by a low-rank matrix in the l_2-norm, is one of the most popular methods for preprocessing data. However, this method can be sensitive to outliers and missing data because the cost function based on the l_2-norm can magnify their influence. Therefore, l_2-based low-rank approximations may find projections that are far from the desired solution. Unlike the l_2-norm, the l_1-norm is more robust to outliers and missing data in statistical estimation. A minimization problem based on the l_1-norm can be regarded as a maximum-likelihood estimation problem under the Laplacian noise distribution [2].

We first consider an approximation problem for vector $\mathbf{y} = (y_1, y_2, \ldots, y_m)^T$ by a multiplication of vector $\boldsymbol{x} \in \mathbb{R}^m$ and scalar α

$$\boldsymbol{y} = \alpha \boldsymbol{x} + \boldsymbol{\delta} \tag{1}$$

where $\boldsymbol{\delta}$ is a noise vector whose elements have independently and identically distributed Laplacian distribution [2]. The probability model for (1) can be written as

$$p(\boldsymbol{y}|\boldsymbol{x}) \sim \exp\left(-\frac{\|\boldsymbol{y} - \alpha \boldsymbol{x}\|_1}{s}\right) \tag{2}$$

where $\|\cdot\|_1$ denotes the l_1-norm, and $s > 0$ is a scaling constant [3]. We assume for a moment that $\boldsymbol{x}$ is given. In this case, maximizing the log likelihood of the observed data is equivalent to minimizing the following cost function:

$$J(\alpha) = \|\boldsymbol{y} - \alpha \boldsymbol{x}\|_1. \tag{3}$$

Then, this minimization problem can be written as

$$J(\alpha) = \sum_i |y_i - \alpha x_i| = \sum_i |x_i| \left|\frac{y_i}{x_i} - \alpha\right|. \tag{4}$$

The global optimal solution for (4) can be found by the weighted median technique for set $\{y_i/x_i | i = 1, \ldots, m\}$ with the ith weight $|x_i|$ [2], [21]. If $x_i = 0$, then the corresponding ith term is ignored because it has no effect on finding the solution.

The approximation problem of (1) can be generalized to the problem of matrix approximation. Let us consider the l_1 approximation of matrix Y such that

$$\min_{P,X} J(P, X) = |Y - PX\|_1 \tag{5}$$

where $Y \in \mathbb{R}^{m \times n}$, $P \in \mathbb{R}^{m \times r}$, and $X \in \mathbb{R}^{r \times n}$ are the observation, projection, and coefficient matrices, respectively.

Here, r is a predefined parameter less than $\min(m, n)$, and $||\cdot||_1$ in (5) is the entry-wise l_1-norm, i.e., $\|Y\|_1 = \sum_{ij} |y_{ij}|$ where y_{ij} is the (i, j)th element of Y, which is different from the induced l_1-norm. In general, (5) is a nonconvex problem, because both P and X are unknown variables. Ke and Kanade [2], [3] propose two ways of solving this problem, one by applying alternating convex minimization and the other by applying alternating weighted median approaches. However, the proposed weighted median method computes the subspace bases column by column, and therefore is potentially more likely to be trapped into a bad local minimum [2]. They prefer the convex programming method, which is more efficient than the weighted median method. Minimizing the cost function over one matrix while keeping the other fixed, and then alternately exchanging roles of the matrices, enables the optimization process to be performed rather efficiently. Such minimization techniques based on alternating iterations have been widely used in subspace analysis [1]–[3], [22] and can be written as

$$\begin{aligned} P^{(t)} &= \arg\min_P \|Y - PX^{(t-1)}\|_1 \\ X^{(t)} &= \arg\min_X \|Y - P^{(t)}X\|_1 \end{aligned} \tag{6}$$

where the superscript t denotes the iteration number.

However, these alternating approaches are still computationally very expensive and require a large memory when the matrix dimension is large. In the following two sections, we propose two novel efficient methods that find the solution for (5) at a much reduced computational cost.

III. Proposed Method: l_1-ARG$_A$

As mentioned previously, solving an l_1-norm-based problem using linear or quadratic programming requires too much time and memory for practical problems. To overcome this, we propose a novel low-rank matrix approximation method.

A. Gradient-Based Update

We first describe the problem of low-rank matrix approximation in the l_1-norm by an alternating gradient descent framework. Let us rewrite the cost function for matrix approximation in (5)

$$\min_{P,X} \quad J(P, X) = \|Y - PX\|_1. \tag{7}$$

Since $|x|$ is not differentiable, we approximate $|x|$ by $\lim_{\epsilon\to 0}\sqrt{x^2+\epsilon^2}$. Then, we approximate the derivative of $|x|$ using the derivative of $\lim_{\epsilon\to 0}\sqrt{x^2+\epsilon^2}$ as follows:

$$\frac{d|x|}{dx} \approx \lim_{\varepsilon\to 0}\frac{\partial\sqrt{x^2+\varepsilon^2}}{\partial x} = \lim_{\varepsilon\to 0}\frac{x}{\sqrt{x^2+\varepsilon^2}} = \operatorname{sgn}(x) \tag{8}$$

where $\operatorname{sgn}(x)$ is the signum function of x and the approximation is exact except at $x = 0$. In this way, we can differentiate (7) with respect to X and find that its derivative is

$$\nabla_X J(P, X) = -P^T \operatorname{sgn}(Y - PX). \tag{9}$$

Here, $\operatorname{sgn}(Y)$ for matrix Y represents a matrix whose (i, j)th element is $\operatorname{sgn}(y_{ij})$.

Now, we consider the problem of finding an optimal step size $\alpha > 0$ to update X by the steepest gradient descent method

$$\begin{aligned} \min_\alpha J(\alpha|P, X, \nabla_X J) &= \|Y - P(X - \alpha\nabla_X J(P, X))\|_1 \\ &= \|Y' - \alpha PP^T \operatorname{sgn}(Y')\|_1 \\ &= \|Y' - \alpha A\|_1 \end{aligned} \tag{10}$$

where $Y' = Y - PX$ and $A = PP^T \operatorname{sgn}(Y')$. We apply the weighted median algorithm to the ratio y'_{ij}/a_{ij} with weight $|a_{ij}|$ to get the step size α that minimizes the cost function (10). Note that in this algorithm, we apply the weighted median algorithm to update either P or X at a time, to reduce the total computation time and this is different from Ke and Kanade [3], where the algorithm is applied column-wise. Finally, Y' and X are updated as

$$\begin{aligned} Y' &\leftarrow Y' - \alpha PP^T \operatorname{sgn}(Y') \\ X &\leftarrow X + \alpha P^T \operatorname{sgn}(Y'). \end{aligned} \tag{11}$$

For P, we can also differentiate (7) with respect to P in the same manner as

$$\nabla_P J(P, X) = -\operatorname{sgn}(Y - PX)X^T. \tag{12}$$

The projection and coefficient matrices P and X are updated alternatingly until convergence is achieved.

However, a serious issue arises in this updating procedure, because there are numerous pairs of P and X that give the same multiplication result of PX. To see this, let us reexamine the minimization problem (7). If $P' = PH^{-1}$ and $X' = HX$ for some nonsingular matrix $H \in \mathbb{R}^{r\times r}$, then

$$\min_{P',X'} \quad J(X', P') = \|Y - P'X'\|_1 = \|Y - PX\|_1. \tag{13}$$

Accordingly, the step size problem for X' can be written as

$$\min_\beta J(\beta|P', X', \nabla_{X'} J) = \|Y' - \beta P'P'^T \operatorname{sgn}(Y')\|_1 \tag{14}$$

where β is a step size. When H is orthogonal, (10) and (14) are the same because of the relation $P'P'^T = PH^{-1}H^{-T}P^T = PH^THP^T = PP^T$. If it is not the case, then the update direction of (14) changes depending on H, i.e.,

$$PP^T \operatorname{sgn}(Y') \neq P'P'^T \operatorname{sgn}(Y'). \tag{15}$$

This means that the update direction depends on the choice of P and X. Therefore, it is important to find P and X that will give a good update direction for fast convergence.

B. Finding an Optimal Direction for Alternating Updates

In the previous section, we have shown that the update direction depends on the representation of P and X, which can influence the convergence rate. This happens because P and X are the intermediate variables of the following basic problem:

$$\begin{aligned} \min_G \quad & \|Y - G\|_1 \\ \text{s.t.} \quad & G \in \mathbb{R}_r^{m\times n} \end{aligned} \tag{16}$$

where $\mathbb{R}_r^{m\times n}$ is a set of $m \times n$ matrices with rank r. However, this problem is difficult to solve directly because $\mathbb{R}_r^{m\times n}$ is not convex. This is why it is common to use alternating updates based on intermediate variables like P and X for low-rank

matrix approximation. In summary, it is difficult to solve the problem (16), while the less difficult problem (7) can still lead to a slow convergence because of the ambiguity of the update direction.

Then, how do we compromise? To answer this question, notice that the gradient with respect to X can also be expressed as the solution to the following problem:

$$\begin{aligned} \min_{\Delta X'} \quad & J(\Delta X'|P, X) = \|Y - P(X + \Delta X')\|_1 \\ \text{s.t.} \quad & \|\Delta X'\|_F^2 = \breve{\epsilon}^2 \end{aligned} \tag{17}$$

where $\Delta X'$ is the variation of X that we are seeking and $\breve{\epsilon} \ll 1$. This problem is to minimize the directional derivative of the cost function with respect to $\Delta X'$ and the optimal $\Delta X'$ is the same as $\nabla_X J$ up to scale if $\breve{\epsilon} \to 0$. To avoid the ambiguity in representing P and X, and to convert the problem as if it were to be solved for $G \in R_r^{m\times n}$ in the basic problem, we modify the constraint as

$$\begin{aligned} \min_{\Delta X'} \quad & J(\Delta X'|P, X) = \|Y - P(X + \Delta X')\|_1 \\ & \triangleq \|Y' - \Delta G'\|_1 \\ \text{s.t.} \quad & \|\Delta G'\|_F^2 \triangleq \|P\Delta X'\|_F^2 = \epsilon^2. \end{aligned} \tag{18}$$

In this modified problem, we search the update direction for X, but the new constraint limits the search domain with respect to $\Delta G' = P\Delta X'$, the update of G, instead of $\Delta X'$. In this manner, we can preserve the convexity of the search domain while avoiding the difficulty that arises from the ambiguity in representing P and X.

By introducing a Lagrange multiplier to (18), the resulting Lagrangian is

$$\|Y' - P\Delta X'\|_1 + \frac{\lambda}{2}(\text{tr}(\Delta X'^T P^T P \Delta X') - \epsilon^2) \tag{19}$$

where tr is the trace operator ($\|A\|_F^2 = \text{tr}(A^T A)$). Differentiating (19) with respect to $\Delta X'$ and equating it to zero, we obtain

$$-P^T \operatorname{sgn}(Y' - P\Delta X') + \lambda P^T P \Delta X' = 0$$

which gives

$$\Delta X' = \frac{1}{\lambda} P^+ \operatorname{sgn}(Y' - P\Delta X') \tag{20}$$

where $P^+ = (P^T P)^{-1} P^T$ is the left pseudoinverse of P. By applying (20) to $\|P\Delta X'\|_F^2 = \epsilon^2$, we obtain

$$\frac{1}{\lambda} = \frac{\epsilon}{\|PP^+ \operatorname{sgn}(Y' - P\Delta X')\|_F} \tag{21}$$

and finally

$$\Delta X' = \frac{P^+ \operatorname{sgn}(Y' - P\Delta X')}{\|PP^+ \operatorname{sgn}(Y' - P\Delta X')\|_F} \cdot \epsilon. \tag{22}$$

For an infinitesimal ϵ, the update direction becomes

$$\begin{aligned} \lim_{\epsilon\to 0} \Delta X' &\propto \lim_{\epsilon\to 0} P^+ \operatorname{sgn}(Y' - P\Delta X') \\ &= P^+ \operatorname{sgn}(Y') \triangleq \Delta X. \end{aligned} \tag{23}$$

Note $\lim_{\epsilon\to 0} \operatorname{sgn}(Y' - P\Delta X') = \operatorname{sgn}(Y')$ in (20) because $\lim_{\epsilon\to 0} \Delta X' = 0$ in (22) and we regard sgn as a limit of a smooth function, as defined in (8). Here, we ignore $\|PP^+ \operatorname{sgn}(Y' - P\Delta X')\|_F$ in (22) because we are interested only in the direction, which is denoted as ΔX, and the step size for the update will be found next. Note that, the update direction of the low-rank approximation is given as

$$\Delta G \triangleq P\Delta X = PP^+ \operatorname{sgn}(Y') \tag{24}$$

and this does not change depending on the representation of P and X, i.e., there is no ambiguity in ΔG unlike $PP^T \operatorname{sgn}(Y')$ in (15). With the new update direction ΔG, we revise the step size problem (10) as the following:

$$\min_{\alpha} \|Y' - \alpha\Delta G\|_1 = \|Y' - \alpha PP^+ \operatorname{sgn}(Y')\|_1 \tag{25}$$

where α is determined by the weighted median technique. For updating P, we can obtain ΔP in the same manner under the constraint ($\|\Delta P' X\|_F^2 = \epsilon^2$) as

$$\Delta P = \operatorname{sgn}(Y')X^+ \tag{26}$$

and find the optimal step size as in (25).

There is an observation to be made on this updating rule.

This new update direction is analogous to the Gauss–Newton update direction in the least-squares problem. The Gauss–Newton direction of $\|F(x)\|_F^2$ is given as $-\nabla_x F(x)^+ F(x)$. If we regard $F(x)$ as a result of $\partial\|F(x)\|_F^2/\partial F(x)$ ignoring its scale, then it is similar to the expression $\Delta X = P^+ \operatorname{sgn}(Y')$. Hence, we may consider this update direction as an extension of the Gauss–Newton method to l_1-norm problems and expect it to be better than the normal gradient direction.

Note that, this procedure is equivalent to changing the representation of P and X so that the fixed matrix, either P or X, is orthonormal. This means that the step size problem (14) of the normal gradient method becomes the same as (25) when P and X are chosen so that P is orthogonal. We can easily find such an orthogonal matrix using the QR decomposition. We call this as the rectified representation. Hence, it is better to use ordinary gradient descent in conjunction with this representation change, which is faster than calculating a pseudoinverse.

C. Summary of the Proposed Algorithm

First, we update P while X is fixed in (7). To make X orthonormal, we apply QR decomposition to X^T

$$\begin{aligned} X^T &= X'^T R \\ PX &= PR^T X' = P'X' \end{aligned} \tag{27}$$

where orthogonal matrix X'^T and upper triangular matrix R are obtained from QR decomposition, and $P' = PR^T$. Then, we can compute ΔP using X' and find the optimal step size using the weighted median algorithm.

Once the update of P is finished, we update X with P fixed. Again, we apply QR decomposition to P to change the representation. The update rule is similar to that of the P update. Then, we continue to update P and X alternatingly; the overall procedure is described in Algorithm 1. We call the method as l_1-norm-based alternating rectified gradient

Algorithm 1 l_1-Norm-Based Matrix Approximation Using the Approximated Alternating Rectified Gradient Method (l_1-ARG$_A$)

1: Input: $Y \in \mathbb{R}^{m\times n}$, the subspace dimension r
2: Output: $P \in \mathbb{R}^{m\times r}$, $X \in \mathbb{R}^{r\times n}$
3: Initialize P to a zero matrix and X randomly
4: $Y' \leftarrow Y$
5: **while** residual Y' does not converge **do**
6: ## P update (Fix X, update P)
7: **while** residual Y' does not converge **do**
8: $X'^T R \leftarrow X^T$, $P' \leftarrow PR^T$
9: $\Delta P \leftarrow \mathrm{sgn}'(Y')X'^T$
10: $(Y', P') \leftarrow \mathrm{Update}(Y', P', X', \Delta P)$
11: **end while**
12: ## X update (Fix P, update X)
13: **while** residual Y' does not converge **do**
14: $PR \leftarrow P'$, $X \leftarrow RX'$
15: $\Delta X \leftarrow P^T \mathrm{sgn}'(Y')$
16: $(Y'^T, X'^T) \leftarrow \mathrm{Update}(Y'^T, X^T, P^T, \Delta X^T)$
17: **end while**
18: **end while**

Algorithm 2 Function: Update (Y, U, V, Z)

1: Input: Y, U, V, Z: matrices
2: Output: T, R: matrices
3: ## Line-search (by weighted median)
4: $\alpha \leftarrow \arg\min_\alpha \|Y - \alpha ZV\|_1$
5: $T \leftarrow Y - \alpha ZV$
6: $R \leftarrow U + \alpha Z$

method based on approximation, l_1-ARG$_A$, because it find the gradient by approximated manner. In the algorithm, P and X are rectified by the QR decomposition at lines 8 and 14, respectively.

To deal with numerical errors, we modify the signum function as

$$\mathrm{sgn}'(x) = \begin{cases} 1 & x \geq \gamma \\ 0 & -\gamma < x < \gamma \\ -1 & x \leq -\gamma \end{cases} \tag{28}$$

where γ is a threshold with a small positive value. Using this modified function, we can find a better solution despite the difficulties that numerical errors might create.

In Algorithm 1, the update of either P or X is repeated until convergence, and then the roles of the matrices are switched. Even though the algorithm can work by just alternating the updates of P and X one by one, the present approach gave us a better performance in some of the experiments, such as the nonrigid motion estimation in Section V-D. This is not exactly an alternating update, but we still call it alternating rectified gradient method. The projection and coefficient matrices are updated by line-search technique using the weighted median method in Algorithm 2.

As mentioned earlier, the step size α is determined using the weighted median algorithm. For the weighted median algorithm, we may use a divide and conquer algorithm such as quick-select [23], [24], which can find the solution in linear time on average. However, in practice, it is faster to use existing sorting functions when the number of elements is not large. Moreover, since we are applying the weighted median algorithm to find the step size, which does not need to be accurate, it is better to calculate the weighted median of randomly selected samples, when the number of samples is large. To see how the weighted median depends on the number of samples, we consider the problem of finding an approximate weighted median from a set consisting of an infinite number of elements. To simplify the problem, we assume that elements have the same weights. Then, the cumulative probability $F(q; 2d+1)$ that the sample median of $2d+1$ samples is less than the $(100 \times q)\%$ quantile of original elements is equal to the cumulative probability that the success is no more than d for a binomial distribution $B(2d+1, 1-q)$. Since the cumulative distribution function of a binomial distribution can be represented in terms of the regularized incomplete beta function, the result is given as

$$F(q; 2d+1) = P(Z \leq d) = I_q(d+1, d+1) \tag{29}$$

where Z is the binomial random variable and I_q is the regularized incomplete beta function. This expression can be calculated numerically, and we have found that

$$F\left(\frac{1}{2} + 0.005; 10^5 + 1\right) - F\left(\frac{1}{2} - 0.005; 10^5 + 1\right) \approx 0.998.$$

This means that if we use 10^5 samples, then the sample median resides within the $\pm 0.5\%$ range of the true median with probability 0.998. Even if this result applies for the case of equally weighted samples, the result is also meaningful for the weighted median if the weights are moderately distributed. This is a valid assumption because ΔG, which is an orthogonal projection of $\mathrm{sgn}'(Y')$, is bounded by $\|\mathrm{sgn}'(Y')\|_F$. In the experiments in Section V, we randomly selected 10^5 samples if the number of elements is greater than 10^5, and then applied an existing sorting function to find the weighted median. There is a small chance that the weighted median technique may not reduce the cost function due to random sampling, but the problem can be resolved by a slight tweak in the algorithm, such as repeating the random sampling until it reduces the cost function.

The downside of the proposed algorithm is the difficulty of guaranteeing whether $P^{(t)}$ and $X^{(t)}$ will converge to a local minimum, due primarily to the assumption that the derivative of $|x|$ is $\mathrm{sgn}(x)$, which is in fact not differentiable at 0. Hence, there is a possibility that the algorithm may find an update direction that does not decrease the cost function when many of the elements of Y' are zero, even though it is not a local minimum. In that case, the step size will be zero and the algorithm will be terminated. Nonetheless, if this happens, it will be near a local minimum since many of the residual elements are zero. Besides, there is usually some Gaussian noise in Y for practical problems, which prevent many of the residual elements from being zero at the same time. Therefore, the proposed algorithm will work well in practical problems and we verify the convergence using real world problems in Section V.

D. Weighted Method of l_1-ARG$_A$ With Missing Data

In real applications, there are not only outliers but also missing data, which can have a negative effect on vision and recognition systems. We solve the problem of low-rank matrix approximation using the l_1-norm in the presence of missing data, which is also known as a matrix completion (MC) problem by extending the result from the previous section.

The problem can be formulated as

$$\min_{P,X} \quad J(P, X|W) = \|W \odot (Y - PX)\|_1 \tag{30}$$

where $\odot$ is the component-wise multiplication or Hadamard product. Here, $W \in \mathbb{R}^{m\times n}$ is a weight matrix, whose element w_{ij} is 1 if y_{ij} is known, and is 0 if y_{ij} is unknown. Similar to the problem (18), we can formulate the weighted low-rank matrix factorization in the l_1-norm under the constraint $\|P\Delta X'\|_F^2 = \epsilon^2$ as

$$\begin{aligned} \min_{\Delta X'} \quad & J(\Delta X'|P, X, W) = \|(W \odot (Y' - P\Delta X'))\|_1 \\ \text{s.t.} \quad & \|P\Delta X'\|_F^2 = \epsilon^2. \end{aligned} \tag{31}$$

Similarly as in Section III-B, the solution to this problem can be represented in vector form as

$$\begin{aligned} \text{vec}(\Delta X) &= (I \otimes P^+)\overline{W}\text{vec}(\text{sgn}(W \odot Y')) \\ &= (I \otimes P^+)\text{vec}(W \odot \text{sgn}(W \odot Y')) \end{aligned} \tag{32}$$

where $\otimes$ is the Kronecker product, $\overline{W} = \text{diag}(\overline{\boldsymbol{w}}) \in \mathbb{R}^{mn\times mn}$, $\overline{\boldsymbol{w}} = (\boldsymbol{w}_1^T, \boldsymbol{w}_2^T, \ldots, \boldsymbol{w}_n^T)^T \in \mathbb{R}^{mn\times 1}$, $\boldsymbol{w}_i$ is the ith column vector of W, and I denotes an $n \times n$ identity matrix. Because the elements of W are either 0 or 1, (32) can be rewritten as

$$\begin{aligned} \text{vec}(\Delta X) &= (I \otimes P^+)\text{vec}(\text{sgn}(W \odot Y')) \\ &= \text{vec}(P^+ \,\text{sgn}(W \odot Y')) \end{aligned} \tag{33}$$

and this gives

$$\Delta X = P^+ \,\text{sgn}(W \odot Y'). \tag{34}$$

Similar to (25), the cost function to find the step size α becomes

$$\begin{aligned} \min_{\alpha} J(\alpha|P, X, W, \Delta X) &= \min_{\alpha} \|W \odot (Y' - \alpha P\Delta X)\|_1 \\ &= \min_{\alpha} \|W \odot Y' - \alpha W \odot (PP^+ \,\text{sgn}(W \odot Y'))\|_1. \end{aligned} \tag{35}$$

Compared with (25), the only difference is the presence of W in the cost function.

When we vary P for a fixed X, we can obtain ΔP and the cost function to find the optimal step size similarly

$$\Delta P = \text{sgn}(W \odot Y')X^+ \tag{36}$$

$$\begin{aligned} \min_{\alpha'} J(\alpha'|P, X, W, \Delta P) &= \min_{\alpha'} \|W \odot (Y' - \alpha'\Delta PX)\|_1 \\ &= \min_{\alpha'} \|W \odot Y' - \alpha' W \odot (\text{sgn}(W \odot Y')X^+X)\|_1. \end{aligned} \tag{37}$$

The step sizes in (35) and (37) can also be solved by the weighted median algorithm.

IV. Proposed Method: l_1-ARG$_D$

A. l_1-ARG$_D$ in the Presence of Outliers

In this section, we propose a second novel method to find a proper descending direction without the gradient approximation of ΔX. Since it is difficult to guarantee that l_1-ARG$_A$ converges to a local minimum, we propose the second novel method with a convergence guarantee. We refer to the algorithm as a l_1-norm-based alternating rectified gradient method using the dual problem, l_1-ARG$_D$. As mentioned earlier, the problem to find the gradient of X for a fixed P in low-rank matrix approximation is formulated as

$$\begin{aligned} \min_{P,X} \quad & \|Y' - P\Delta X\|_1 \\ \text{s.t.} \quad & ||P\Delta X||_F^2 = \epsilon^2. \end{aligned} \tag{38}$$

We reformulate (38) to an unconstrained problem as

$$\min_{\Delta X} \quad f_\eta(X, \Delta X) \triangleq ||Y' - P\Delta X||_1 + \frac{1}{2\eta}||P\Delta X||_F^2 \tag{39}$$

where $\eta > 0$ is a weight parameter. Here, we assume that P is orthonormalized using the QR decomposition, i.e., $||P\Delta X||_F^2 = ||\Delta X||_F^2$.

We can obtain the Lagrangian of (39) by substituting $||Y' - P\Delta X||_1$ to Z as

$$\begin{aligned} \mathcal{L}(\Delta X, \Lambda, M) = \mathbf{1}^T Z\mathbf{1} + \frac{1}{2\eta}||\Delta X||_F^2 + \text{tr}(\Lambda^T(Y' - P\Delta X - Z)) \\ + \text{tr}(M^T(-Y + P\Delta X - Z)) \end{aligned} \tag{40}$$

where $\mathbf{1} \in \mathbb{R}^m$ and $\Lambda, M \leq 0$ are Lagrange multipliers. By taking a derivative of (40) and solving for Z and ΔX at a stationary point, we can obtain $\mathbf{1}\mathbf{1}^T - \Lambda - M = 0$ and $\Delta X = \eta P^T(\Lambda - M) = \eta P^T\widetilde{V}$, respectively, where $\widetilde{V} \triangleq \Lambda - M$ and $-1 \leq \widetilde{v}_{ij} \leq 1$ for all elements of $\widetilde{V}$. Therefore, (40) can be reformulated as

$$\begin{aligned} & \frac{1}{2\eta}||\Delta X||_F^2 + \text{tr}((\widetilde{V})^T(Y' - P\Delta X)) \\ & \text{s.t.} \quad -1 \leq \widetilde{v}_{ij} \leq 1. \end{aligned} \tag{41}$$

Hence, the dual problem of (39) is constructed using the corresponding primal solution $\Delta X = \eta P^T\widetilde{V}$ and $\eta\widetilde{V} = V$ as

$$\begin{aligned} \max_{V} \quad & g_\eta(V) \triangleq \frac{1}{\eta}\text{tr}(V^TY') - \frac{1}{2\eta}||P^TV||_F^2 \\ \text{s.t.} \quad & -\eta \leq v_{ij} \leq \eta. \end{aligned} \tag{42}$$

We use the proximal gradient technique [16] to solve this problem. We convert the sign of (42) and reformulate it as an unconstrained problem

$$\min_{V} \quad -\frac{1}{\eta}\text{tr}(V^TY') + \frac{1}{2\eta}||P^TV||_F^2 + I_\eta(V) \tag{43}$$

where $I_\eta(V)$ is the indicator function for each element of matrix V

$$I_\eta(v_{ij}) = \begin{cases} 0 & -\eta \leq v_{ij} \leq \eta \\ \infty & \text{else.} \end{cases} \tag{44}$$

Denoting U as the V in the previous step, the proximal approximation [16] of (43) is given as

$$\frac{1}{\eta}\operatorname{tr}((V-U)^T(-Y'+PP^TU))+\frac{L}{2\eta}||V-U||_F^2 + \frac{1}{2\eta}||P^TU||_F^2-\frac{1}{\eta}\operatorname{tr}(U^TY')+I_\eta(V) \quad (45)$$

where L is the Lipschitz constant of (43) and is 1 in this case because P is orthogonal.

Equation (45) can be simplified as

$$\frac{1}{2\eta}||V-U-Y'+PP^TU||_F^2+I_\eta(V)+\text{constant} \quad (46)$$

and this gives the following result:

$$V=\begin{cases}\eta & V'>\eta\\ V' & -\eta<V'<\eta\\ -\eta & V'<-\eta\end{cases} \quad (47)$$

where

$$V'=Y'+U-PP^TU. \quad (48)$$

Since this iterative process itself can take a nonignorable amount of time, we perform the iteration just enough to find a good descending direction, rather than calculating the exact optimal solution. We update the solution V and corresponding primal solution $\Delta X = P^TV$ until the ratio between the difference of the previous and current primal cost values and the difference of the previous primal and current dual cost values is no less than a positive scalar $0<\beta\le 1$ as

$$\frac{f_\eta(X,\Delta X_k)-f_\eta(X,\Delta X_{k+1})}{f_\eta(X,\Delta X_k)-g_\eta(V_{k+1})}\ge\beta. \quad (49)$$

Let $\Delta X^*=\arg\min_{\Delta X} f_\eta(X,\Delta X)$, then we obtain the following relation:

$$\begin{aligned}f_\eta(X,0)-f_\eta(X,\Delta X)&\ge\beta(f_\eta(X,0)-g_\eta(V))\\&\ge\beta(f_\eta(X,0)-f_\eta(X,\Delta X^*)).\end{aligned} \quad (50)$$

Note that during the proximal optimization, $g_\eta(V_{k+1})$ is always not larger than $f_\eta(X,\Delta X_{k+1})$. After finding a solution that satisfies (49), we apply the weighted median method as an exact line-search[1] to find the optimal step size of the gradient. The overall procedure is described in Algorithm 3. In the algorithm, η is decreased during the iteration and is bounded by $0<\eta_{\min}\le\eta\le\eta_{\max}<\infty$ where $\eta_{\min}$ and $\eta_{\max}$ are predefined constants. P and X are rectified by the QR decomposition at lines 7 and 11 in the algorithm, respectively. We find the gradient of P or X by Algorithm 4.

The main difference between the two proposed methods is that we can formally guarantee that l_1-ARG$_D$ converges to a subspace-wise local minimum (see Section IV-B), whereas a local minimum is not guaranteed for l_1-ARG$_A$ due to the approximation of the l_1 cost function. Although both algorithms may reach similar cost values, they can find different solutions, as shown in Section V.

[1]Here, we assume that an exact line-search is performed to simplify the proof in the below.

Algorithm 3 l_1-Norm-Based Matrix Approximation Using the Exact Alternating Rectified Gradient Method (l_1-ARG$_D$)

1: Input: $Y\in\mathbb{R}^{m\times n}$, low-rank r, $\beta=10^{-4}$, $\eta_{\min}=10^{-6}$
2: Output: $P\in\mathbb{R}^{m\times r}$, $X\in\mathbb{R}^{r\times n}$
3: Initialize P to a zero matrix and X randomly, $\eta=\infty$
4: $Y'\leftarrow Y$
5: **while** residual Y' does not converge **do**
6: # P update (Fix X, update P)
7: $X'^TR\leftarrow X^T$, $P'\leftarrow PR^T$
8: $\Delta P^T\leftarrow$ findGradient(X'^T, P^T, Y'^T, V^T, η, $\eta_{\min}$, β)
9: $(Y',P')\leftarrow$ Update(Y', P', X, ΔP)
10: # X update (Fix P, update X)
11: $PR\leftarrow P'$, $X\leftarrow RX'$
12: $\Delta X\leftarrow$ findGradient(P, X, Y', V, η, β)
13: $(Y'^T,X^T)\leftarrow$ Update(Y'^T, X^T, P^T, ΔX^T)
14: **end while**

Algorithm 4 Function: findGradient (K, L, Y, V, η, $\eta_{\min}$, β)

1: Input: K, L, Y, and V: matrices; η, $\eta_{\min}$, β: scalars
2: Output: ΔS: a matrix
3: Description:
4: $\eta\leftarrow\max(\min(\eta,||Y||_1/mn),\eta_{\min})$, $k=1$, $V_0=0$
5: $f_\eta(K,\Delta K_0)=f_\eta(K,\Delta K_1)=||Y||_1$, $g_\eta(V_1)=0$
6: **while** $\frac{f_\eta(K,\Delta K_{k-1})-f_\eta(K,\Delta K_k)}{f_\eta(K,\Delta K_{k-1})-g_\eta(V_k)}<\beta$ **do**
7: $\eta\leftarrow\max(\frac{\eta}{2},\eta_{\min})$
8: $V_k\leftarrow Y+V_{k-1}-V_{k-1}L^TL$ and by (47)
9: $\Delta K_k\leftarrow V_kL^T$
10: $f_\eta(K,\Delta K_{k+1})\leftarrow||Y-\Delta K_kL||_1+\frac{1}{2\eta}||\Delta K_k||_F^2$
11: $g_\eta(V_{k+1})\leftarrow\operatorname{tr}(Y^TV_k)-\frac{1}{2\eta}||\Delta K_k||_F^2$
12: $k\leftarrow k+1$
13: **end while**
14: $\Delta S\leftarrow\Delta K_{k-1}$

B. Proof of Convergence

Regardless of the initial point, the proposed method, l_1-ARG$_D$, which is a descent algorithm, converges to a subspace-wise local minimum according to the Zangwill's global convergence theorem [25], [26]. Subspace-wise local minimum is defined as follows.

Definition 1 (Subspace-Wise Local Minimum): Let the cost function of l_1-ARG$_D$ be $J(P,X)\triangleq||Y-PX||_1$. If there is no ΔX or ΔP such that $||Y-P(X+\Delta X)||_1<||Y-PX||_1$ or $||Y-(P+\Delta P)X||_1<||Y-PX||_1$, then (P,X) is a subspace-wise local minimum.

A local minimum is a subspace-wise local minimum. If a cost function is smooth, a subspace-wise local minimum is also a local minimum [26]. However, the cost function (7) is not smooth, and consequently, a subspace-wise local minimum may not be a local minimum. Nonetheless, it is worth finding a subspace-wise local minimum because a subspace-wise local minimum is a necessary condition to be a local minimum. It also minimizes the cost function as well as the other state-of-the-art methods in the experiments of Section V.

Let us denote $A:(\mathcal{P},\mathcal{X})\rightarrow(\mathcal{P},\mathcal{X})$ as a point-to-set mapping [25], [26] that describes the behavior of l_1-ARG$_D$,

where $\mathcal{P}$ and $\mathcal{X}$ are the domains of P and X, respectively. According to the Zangwill's theorem, a descent algorithm is globally convergent under the following three conditions (converges to a subspace-wise local minimum irrespective of the initial point).

1) All (P_k, X_k) should be contained in a compact set.
2) For cost function $J(P, X) = ||Y - PX||_1$:
   a) if (P, X) is not in the solution set consisting of subspace-wise local minimums, $J(P', X') < J(P, X)$ for all $(P', X') \in A(P, X)$.
   b) if (P, X) is in the solution set, $J(P', X') \leq J(P, X)$ for all $(P', X') \in A(P, X)$.
3) Mapping A is closed at points that are not subspace-wise local minimum.

We will show that l_1-ARG$_D$ satisfies these conditions to prove its global convergence. We prove the conditions only for the case of updating X while P is orthogonal, without loss of generality, and the condition for updating P can be proved similarly.

Proposition 1: The sequence (P_k, X_k) produced by l_1-ARG$_D$ is contained in a compact set.

Proof: Since l_1-ARG$_D$ is a descent algorithm, it only chooses a point that does not increase the cost function, and always satisfies the relation $||Y - P_k X_k||_1 \leq ||Y||_1$ for an appropriate choice of P_0 and X_0. Since P_k is orthogonal

$$\begin{aligned} ||Y||_1^2 &\geq ||Y - P_k X_k||_1^2 \geq ||Y - P_k X_k||_F^2 \\ &\geq (||Y||_F - ||P_k X_k||_F)^2 = (||Y||_F - ||X_k||_F)^2. \end{aligned} \quad (51)$$

From this, we obtain the following relation:

$$||Y||_F - ||Y||_1 \leq ||X_k||_F \leq ||Y||_F + ||Y||_1. \quad (52)$$

Therefore, X_k is contained in a bounded and closed set, i.e., a compact set. Similarly, we can show P_k is contained in a compact set. Therefore, (P_k, X_k) is contained in a compact set. ∎

Condition 2) can also be proved as follows.

Proposition 2: $J(P_k, X_k)$ is strictly decreasing for (P_k, X_k) that is not subspace-wise local minimum.

Proof: If (P, X) is not a subspace-wise local minimum, $||Y - P(X + \Delta X)||_1 < ||Y - PX||_1$ for some ΔX. Since $J(P, X)$ is a convex function for a fixed P, the following relation is satisfied for any constant ν, $0 \leq \nu \leq 1$:

$$\begin{aligned} &||Y - P(X + \nu \Delta X)||_1 \\ &\quad \leq (1 - \nu)||Y - PX||_1 + \nu||Y - P(X + \Delta X)||_1. \end{aligned} \quad (53)$$

Now, we consider the following:

$$\begin{aligned} &f_\eta(X, 0) - f_\eta(X, \nu \Delta X) \\ &= ||Y - PX||_1 - ||Y - P(X + \nu \Delta X)||_1 - \frac{\nu^2}{2\eta}||\Delta X||_F^2 \\ &\geq ||Y - PX||_1 - (1 - \nu)||Y - PX||_1 \\ &\quad - \nu||Y - P(X + \Delta X)||_1 - \frac{\nu^2}{2\eta}||\Delta X||_F^2 \\ &= \nu\{||Y - PX||_1 - ||Y - P(X + \Delta X)||_1\} - \frac{\nu^2}{2\eta}||\Delta X||_F^2 \\ &= \nu a_1 - \frac{\nu^2}{2} a_2 \end{aligned} \quad (54)$$

where $a_1 = ||Y - PX||_1 - ||Y - P(X + \Delta X)||_1$ and $a_2 = \frac{1}{\eta}||\Delta X||_F^2$. If $0 < \nu < \frac{2a_1}{a_2}$, $f_\eta(X, 0) - f_\eta(X, \nu \Delta X)$ is larger than 0, which means that there exists $\nu \Delta X$ that satisfies $f_\eta(X, 0) > f_\eta(X, \nu \Delta X) \geq f_\eta(X, \Delta X^*)$. Therefore, according to (50), l_1-ARG$_D$ will find a direction $\Delta X' (= \nu \Delta X)$ that satisfies

$$\begin{aligned} &f_\eta(X, 0) - f_\eta(X, \Delta X') \geq \beta(f_\eta(X, 0) - f_\eta(X, \Delta X^*)) > 0 \\ &||Y - PX||_1 > ||Y - P(X + \Delta X')||_1 + \frac{1}{2\eta}||\Delta X'||_F^2 \end{aligned} \quad (55)$$

which is a strictly descending direction when (P_k, X_k) is not in the solution set. ∎

Now, to prove the condition 3), we first show that ΔX^* is a continuous function with respect to X and η.

Proposition 3: If $X_k \to \overline{X}$ and $\eta_k \to \overline{\eta}$, then $\Delta X_k^* \to \Delta \overline{X}^* = \arg\min_{\Delta X} f_{\overline{\eta}}(\overline{X}, \Delta X)$.

Proof: We first state some facts to prove the proposition. First, the optimal sequence $\{\Delta X_k^*\}$ is obviously contained in a bounded and closed set

$$\begin{aligned} \frac{1}{2\eta_k}||\Delta X_k^*||_F^2 &\leq f_{\eta_k}(X_k, \Delta X_k^*) \leq f_{\eta_k}(X_k, 0) \\ &= ||Y - PX_k||_1 \leq ||Y||_1. \end{aligned} \quad (56)$$

(This can also be deduced from the fact that the domain of X_k is compact.) Second, ΔX_k^* satisfies the relation $f_{\eta_k}(X_k, \Delta X_k^*) \leq f_{\eta_k}(X_k, \Delta X)$ for any ΔX, which is the very definition of ΔX_k^*. Third, $f_\eta(X, \Delta X)$ is a strictly convex function with respect to ΔX for a given (X, η) because of the term $\frac{1}{2\eta}||\Delta X||_F^2$. Hence, $f_\eta(X, \Delta X)$ has a unique optimal ΔX^*. Since $\{\Delta X_k^*\}$ is bounded, there must exist a convergent subsequence $\{\Delta X_{k_n}^*\}$, i.e., $\Delta X_{k_n}^* \to \Delta \breve{X}$. Then, for any ΔX, we can obtain the following relation:

$$\begin{aligned} f_{\overline{\eta}}(\overline{X}, \Delta \breve{X}) &= \lim_{n \to \infty} f_{\eta_{k_n}}(X_{k_n}, \Delta X_{k_n}^*) \\ &\leq \lim_{n \to \infty} f_{\eta_{k_n}}(X_{k_n}, \Delta X) = f_{\overline{\eta}}(\overline{X}, \Delta X). \end{aligned} \quad (57)$$

The only $\Delta \breve{X}$ that satisfies the relation is $\Delta \overline{X}^*$. Thus, any convergent subsequence of $\{\Delta X_k^*\}$ has the same limit $\Delta \overline{X}^*$. Since ΔX_k^* is bounded and all the convergent subsequences has the same limit, ΔX_k^* converges to the limit $\Delta \overline{X}^*$. ∎

Next, we define a function $K(X, \eta, \Delta X)$ assuming that X is not a local minimum

$$K(X, \eta, \Delta X) \triangleq \frac{f_\eta(X, 0) - f_\eta(X, \Delta X)}{f_\eta(X, 0) - f_\eta(X, \Delta X^*)}. \quad (58)$$

Proposition 4: $K(X, \eta, \Delta X)$ is continuous for nonlocal-minimum X.

Proof: $K(X, \eta, \Delta X)$ is composed of $f_\eta(X, 0)$, $f_\eta(X, \Delta X)$, and $f_\eta(X, \Delta X^*)$ with subtraction and division operations. Also, $f_\eta(X, 0)$ and $f_\eta(X, \Delta X)$ are continuous functions with respect to X, ΔX, and η ($\eta_{\min} \leq \eta \leq \eta_{\max}$), and so is $f_\eta(X, \Delta X^*)$ by Proposition 3. Moreover, $f_\eta(X, 0) > f_\eta(X, \Delta X^*)$ when X is not a local minimum. Therefore $K(X, \eta, \Delta X)$ is also continuous. ∎

Now finally, we prove that l_1-ARG$_D$ satisfies condition 3). Since l_1-ARG$_D$ uses an exact line-search, which is a closed mapping [26], we only need to prove that the procedure for finding a descent direction is a closed mapping at a

TABLE I
PERFORMANCE OF THE PROPOSED METHODS WITH/WITHOUT APPLYING RECTIFICATION

	m=n=500, r=40			m=n=1,000, r=80			m=n=2,000, r=160		
Algorithm	Error	Time (sec)	Iterations	Error	Time (sec)	Iterations	Error	Time (sec)	Iterations
l_1-ARG$_A$	0.867	1.1 ± 0.0	6 ± 0.0	0.869	2.8 ± 0.1	6.1 ± 0.3	0.869	10.0 ± 0.1	6 ± 0.0
l_1-ARG$_A$(no QR)	0.870	6.5 ± 0.7	36.7 ± 3.7	0.871	16.6 ± 0.7	36.4 ± 3.9	0.872	60.2 ± 1.1	35.6 ± 1.5
l_1-ARG$_D$	0.868	0.7 ± 0.2	19.2 ± 4.5	0.869	1.8 ± 0.1	15.2 ± 0.9	0.869	7.6 ± 0.5	15.3 ± 1.6
l_1-ARG$_D$(no QR)	0.870	157.8 ± 25.1	365.1 ± 45.1	0.871	970.3 ± 60.0	358.1 ± 16.4	0.871	5526.7 ± 240.5	365.2 ± 13.7

nonlocal minimum. To do this, we define two point-to-set mappings G and H. $\Delta X \in G(X, \eta)$ determines the descending direction, and $\eta' \in H(\eta)$ determines η, where η' is the value of η in the next iteration. $H(\eta)$ is defined as $H(\eta) = [\eta_{\min}, \eta_{\max}]$ (η' is determined independently, regardless of η), and $G(X, \eta)$ is defined as

$$G(X, \eta) = \{\Delta X | f_\eta(X, 0) - f_\eta(X, \Delta X) \geq \beta(f_\eta(X, 0) - g_\eta(V))\}.$$

If X is not a local minimum, then this is the same as $G(X, \eta) = \{\Delta X | K(X, \eta, \Delta X) \geq \beta\}$.

Proposition 5: Let Q be a point-to-set mapping defined as $(\Delta X, \eta') \in Q(X, \eta)$, where $\Delta X \in G(X, \eta')$ and $\eta' \in H(\eta)$. Then, Q is a closed mapping.

Proof: Here, H is obviously a closed mapping and the domain of η is a bounded set, hence $Q(X, \eta)$, which is a composition of G and H, is a closed mapping if G is a closed mapping. Since K is a continuous function with respect to $(X, \eta, \Delta X)$, $K(\overline{X}, \overline{\eta}, \Delta\overline{X}) = \lim_{k\to\infty} K(X_k, \eta_k, \Delta X_k) \geq \beta$ if $X_k \to \overline{X}$, $\eta_k \to \overline{\eta}$, and $\Delta X_k \to \Delta\overline{X}$. Therefore, G is a closed mapping. ■

Q describes the behavior of finding the descent direction in l_1-ARG$_D$. The proposed method is globally convergent by the proofs for the three conditions. The local convergence rate is hard to find, but we show empirically that l_1-ARG$_D$ gives fast convergence in Section V. Table I shows the comparison between the proposed methods with and without applying rectification (QR decomposition) for three reconstruction problems with 5% outliers over 10 independent runs. As shown in the table, the methods using rectification take much shorter execution time and need less number of iterations, and give lower reconstruction error.

C. Weighted Method of l_1-ARG$_D$ With Missing Data

The proposed method, l_1-ARG$_D$, can be applied to real application problems in the presence of missing data. We solve the problem of low-rank matrix approximation using the l_1-norm by extending the proposed method as a weighted low-rank approximation problem.

The problem can be formulated as

$$||W \odot (Y' - P\Delta X)||_1 + \frac{1}{2\eta}||P\Delta X||_F^2 \tag{59}$$

where η is a small positive constant. We assume that P is orthonormalized by the QR decomposition.

The dual problem of (59) is constructed in the similar fashion as in the previous section

$$\max_V \quad \frac{1}{\eta}\text{tr}((W \odot V)^T Y') - \frac{1}{2\eta}||P^T(W \odot V)||_F^2 \quad \text{s.t.} \quad -\eta \leq V_{ij} \leq \eta \tag{60}$$

and this gives the following unconstrained minimization problem as a proximal mapping operator:

$$\min_V \quad \frac{1}{2\eta}||P^T(W \odot V)||_F^2 - \frac{1}{\eta}\text{tr}((W \odot V)^T Y') + I_\eta(V) \tag{61}$$

where $I_\eta(V)$ is an indicator function. Now, we consider the following approximation of (61):

$$\frac{1}{\eta}\text{tr}((V - U)^T[-W \odot Y' + W \odot (PP^T(W \odot U))]) + \frac{L}{2\eta}||V - U||_F^2 + I_\eta(V) + \text{constant} \tag{62}$$

where L is the Lipschitz constant ($L = 1$). Then, this can be reformulated as

$$\frac{1}{2\eta}||V - U - W \odot Y' + W \odot (PP^T(W \odot U)||_F^2 + I_\eta(V) + \text{constant}. \tag{63}$$

Therefore, we obtain the result in the same form as (47) with $V' = U + Y' + W \odot (PP^T U)$.

V. EXPERIMENTAL RESULTS

We evaluated the performance of the proposed methods (l_1-ARG$_A$ and l_1-ARG$_D$) by experimenting with various data. We compared the proposed algorithms with other methods (Inexact ALM (IALM) and Exact ALM (EALM) [12], ALADM [8], l_1-AQP [3], Regl_1-ALM [10]) in terms of the reconstruction error and execution time. The initial projection and coefficient matrices were set to zero and Gaussian random numbers, respectively, for the proposed methods and l_1-AQP. All the elements of the weight matrix for Regl_1-ALM were set to 1 for nonweighted factorization problems. In addition, the weighted median method used in the proposed methods was implemented, as described in Section III. We set $\rho = 10^{-5}$ in the stopping condition and γ as the same as ρ for all of the proposed methods. The trace-norm regularizer of Regl_1-ALM was set to 20, which gave the best performance in the experiments, if not stated otherwise.

We also performed experiments with missing data using the weighted version of the proposed methods (Wl_1-ARG$_A$

TABLE II
AVERAGE PERFORMANCE FOR 25% OUTLIERS

	m=n=500, r=40		m=n=1,000, r=80		m=n=2,000, r=160		m=n=5,000, r=400		m=n=8,000, r=640		m=n=10,000, r=800	
Algorithm	Error	Time	Error	Time	Error	Time	Error	Time	Error	Time	Error	Time (sec)
l_1-ARG$_A$	5.202	2.719	15.294	9.761	28.595	30.437	83.911	224.023	112.936	685.967	156.335	1657.59
l_1-ARG$_D$	2.583	0.693	6.453	2.002	11.791	8.872	35.678	73.718	49.728	194.967	71.658	387.328
IALM	3.814	2.973	9.371	25.921	21.378	209.978	172.153	3054.72	623.671	19402.41	-	-
EALM	3.375	83.237	7.340	694.947	14.455	5337.55	-	-	-	-	-	-
ALADM	6.141	0.159	19.937	0.848	33.435	4.045	179.167	38.449	236.770	136.919	387.052	447.013
$Regl_1$-ALM	2.757	31.443	5.142	165.049	14.154	926.017	636.373	8760.74	-	-	-	-
l_1-AQP	21.984	4222.22	-	-	-	-	-	-	-	-	-	-

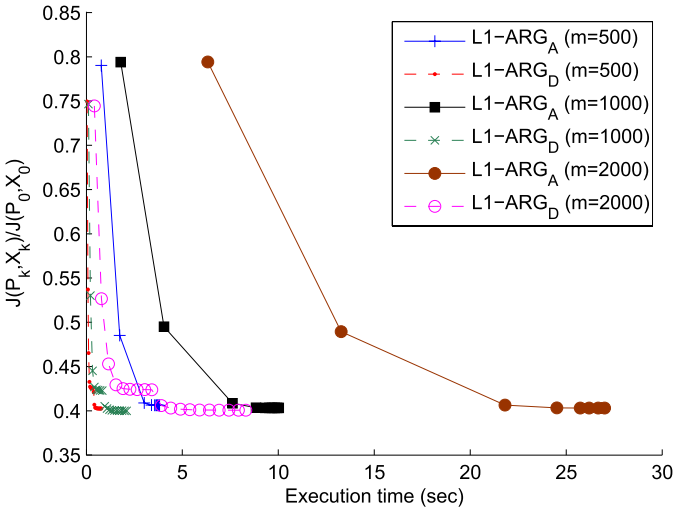


Fig. 1. Normalized cost function of the proposed algorithms for three examples (500×500, 1000×1000, and 2000×2000).

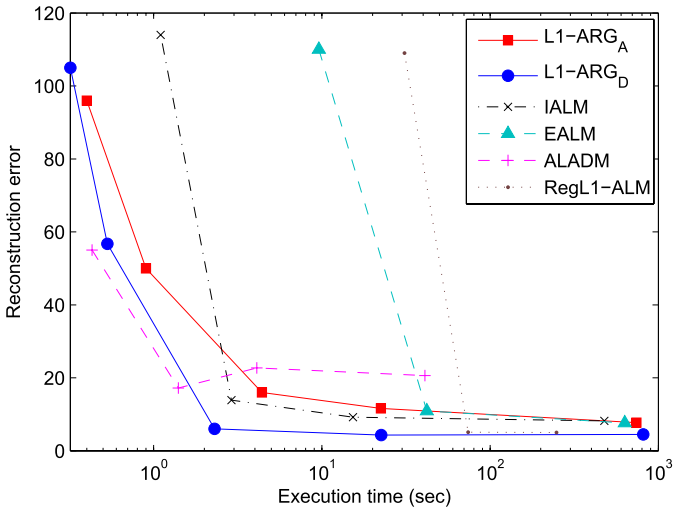


Fig. 2. Reconstruction error as a function of the execution time ($m = 1000$, $n = 1000$, and $r = 80$).

and Wl_1-ARG$_D$) in Sections IV and V, and the performances were compared with those of other methods that can handle missing data (ALADM-MC, which is a matrix completion (MC) version of ALADM [8], Regl_1-ALM [10]). We did not evaluate the methods l_1-AQP [3] for large-scale data because of its heavy computational complexity and memory requirement. We set the parameters of ALADM and ALADM-MC as described in [8], and all of the parameters of the proposed methods were the same as those of nonweighted versions. To show the usefulness of the proposed algorithm, we also applied the proposed methods to the nonrigid SFM problem [10]. All the experiments were conducted using MATLAB on a computer with 8-GB RAM and a 3.4-GHz quad-core CPU.

A. Synthetic Data With Outliers

First, we applied the proposed methods to synthetic examples with outliers. We generated an $(m \times r)$ matrix B and an $(r \times n)$ matrix C whose elements were uniformly distributed in the range $[-1, 1]$. We also generated an $(m \times n)$ noise matrix N whose elements had the Gaussian distribution with zero mean and variance of 0.01. Based on $Y_0 = BC + N$, we constructed the observation matrix Y by replacing 25% of the elements from the 25% randomly selected samples in Y_0 by outliers that were uniformly distributed in the range $[-10, 10]$. We generated six sets from small-size to large-scale examples (500×500–$10\,000 \times 10\,000$). We set the rank of each example matrix to $\min(m, n) \times 0.08$. We compared the proposed methods with IALM, EALM, ALADM, Regl_1-ALM, and l_1-AQP in terms of the reconstruction error and execution time. We used the global parameter for IALM and EALM, as in [12].

In the experiment, the average reconstruction error $E_1(r)$ was calculated as

$$E_1(r) = \frac{1}{n}||Y^{\text{org}} - Y^{\text{low-rank}}||_1 \tag{64}$$

where n is the number of samples, Y^{org} is the ground truth, $Y^{\text{low-rank}}$ is the matrix approximated by an algorithm.

The average reconstruction errors and execution times are shown in Table II. We did not evaluate the methods l_1-AQP, EALM, and Regl_1-ALM for large-scale data because of their heavy computational load. Unlike the fixed-rank

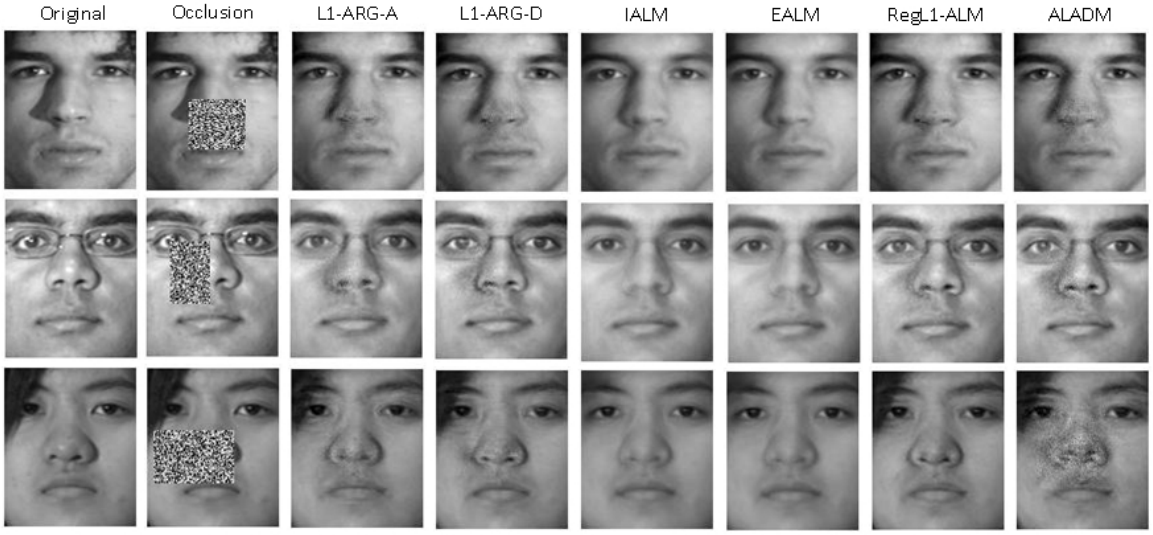


Fig. 3. Face images with occlusions and their reconstructed faces.

TABLE III
RECONSTRUCTION ERROR WITH RESPECT TO VARIOUS r FOR A 1000×1000 MATRIX WITH RANK 80

Algorithm	r=70	r=75	r=80	r=85	r=90
l_1-ARG$_A$	202.74	141.64	14.88	15.08	19.68
l_1-ARG$_D$	188.28	126.03	6.16	23.19	45.03
ALADM	199.76	144.13	17.16	30.61	46.63
$Regl_1$-ALM	193.06	129.19	5.01	12.39	21.39

TABLE IV
AVERAGE PERFORMANCE FOR FACE DATA WITH OCCLUSIONS

	m=12,000, n=830, r=100	
Algorithm	Error (E_1)	Time (sec)
l_1-ARG$_A$	276.957	71.164
l_1-ARG$_D$	279.442	29.760
IALM	261.895	275.426
EALM	257.392	10543.432
$Regl_1$-ALM	287.749	478.168
ALADM	314.298	9.902

approximation methods that give the matrix whose rank is approximately 8% of the original matrix dimension, IALM and EALM give the matrix whose rank is approximately 55% of the original matrix dimension on average in this section. In the table, l_1-ARG$_D$ gives the best result in terms of the reconstruction error and execution time. Although ALADM takes a shorter execution time compared with the proposed methods, it gives a poor reconstruction performance. The proposed methods are superior to the other methods especially for large-scale problems because it uses the weighted median algorithm to handle large-scale problems efficiently. The computational complexities of the proposed methods, ALADM, and l_1-AQP are $O(rmn)$ for each iteration. However, l_1-AQP have to perform a whole convex optimization in each iteration, which is very inefficient in terms of processing time. The computational complexity is $O(\min(m,n)\max(m,n)^2)$ for IALM and EALM, and $O(r\max(m,n)^2)$ for Regl_1-ALM, for each iteration. IALM, EALM, and Regl_1-ALM perform SVD in each iteration, and hence, need much computation time for a large-scale matrix. Fig. 1 shows the cost function of the proposed methods at each iteration for three examples (500×500, 1000×1000, and 2000×2000). As shown in the figure, the cost function of l_1-ARG$_D$ decreases much faster than that of l_1-ARG$_A$, and both methods converge to nearly the same value. Fig. 2 shows the reconstruction error with respect to the execution time for an example (1000×1000). In the figure, the proposed method l_1-ARG$_D$ outperforms other methods. Table III shows the reconstruction error with respect to various r for a 1000×1000 matrix with rank 80. As shown in the table, l_1-ARG$_D$ gives the best results when r is lower than or equal to the exact rank, whereas l_1-ARG$_A$ shows good results when r is larger than the exact rank. It can be explained as follows. Since l_1-ARG$_D$ tries to find a solution that minimizes the cost function for a given r, the performance can be a little bit poorer when r is not correct. l_1-ARG$_A$ finds an approximate solution to the original problem, hence, its result may be worse than l_1-ARG$_D$. However, l_1-ARG$_A$ is less sensitive to the rank uncertainty.

B. Face Reconstruction

We applied various methods to face reconstruction problems and compared their performances. In the experiments, we used 830 images having five different illuminations for 166 people from the Multi-PIE face database [27], which were resized to 100×120 pixels. The intensity of each pixel was normalized to have a value in the range of $[0, 1]$. Each 2-D image was converted to a 12 000-dimensional vector.

TABLE V
AVERAGE PERFORMANCE FOR 20% OUTLIERS AND MISSING DATA

	m=n=500, r=40		m=n=1,000, r=80		m=n=2,000, r=160		m=n=5,000, r=400		m=n=8,000, r=640	
Algorithm	Error	Time	Error	Time	Error	Time	Error	Time	Error	Time (sec)
Wl_1-ARG$_A$	3.966	4.188	8.619	11.565	19.506	43.357	51.087	308.300	76.690	968.353
Wl_1-ARG$_D$	2.384	0.877	4.967	3.116	10.104	13.272	26.197	95.394	40.127	302.015
ALADM-MC	3.214	0.251	10.122	1.184	25.891	5.748	67.453	48.059	92.389	325.493
$Regl_1$-ALM	2.575	9.402	45.074	53.152	69.273	264.823	89.772	2528.845	191.142	8111.565

TABLE VI
AVERAGE PERFORMANCE FOR FACE DATA WITH OCCLUSIONS AND MISSING BLOCKS

	m=12,000, n=830, r=100	
Algorithm	Error (E_1)	Time (sec)
Wl_1-ARG$_A$	305.893	262.976
Wl_1-ARG$_D$	319.462	82.671
ALADM-MC	387.628	11.872
$Regl_1$-ALM	327.556	538.014

We only considered an occlusion case for the experiments of the images and measured the average reconstruction error for occluded images. To generate occlusions, 50% of the images were randomly selected, and each of selected images was occluded by a randomly located rectangle, whose size varied in the range 20×20 pixels–60×60 pixels, with each pixel of the rectangle having a value randomly selected from [0, 1]. We could not apply l_1-AQP and EALM to these problems because they required too much computation time (more than an hour).

Fig. 3 shows some examples of face images with occlusions and their reconstructed faces with 100 projection vectors. In the figure, we can observe that the occlusion blocks have almost disappeared for most of the cases. IALM and EALM tend to produce blurry images, and ALADM gives the poorest results among the methods. Table IV shows the average reconstruction errors E_1 for the face images. In the table, we can observe that our methods show competitive performance in both of the reconstruction error and processing time compared with the other methods. IALM and EALM give a little bit smaller errors than our methods, because the ranks of their reconstructed matrices are higher (around 200) than the others (100). Except ALADM, which gives the poorest reconstruction error, all the compared methods are about four to 350 times slower than our methods.

C. Experiments With Missing Data

We performed experiments with simple examples in the presence of missing data using the proposed methods Wl_1-ARG$_A$ and Wl_1-ARG$_D$ compared with the other state-of-the-art methods, ALADM-MC [8] and Regl_1-ALM [10], which can handle missing data. We generated five examples, as in Section V-A. Here, we did not perform the experiment for a matrix of $10\,000 \times 10\,000$ because of memory limitation. To construct the weight matrix, we randomly selected 20% of the elements of matrix W for each example and set them to zero (missing), while the other elements were set to one.

Table V shows the average result for the five examples with outlier and missing data. In the table, Wl_1-ARG$_D$ gives the best performance and needs much shorter execution time than the other methods except ALADM-MC. Although ALADM-MC gives the shortest execution time, its performance is much worse than the proposed methods. Because of the execution time and the performance, Regl_1-ALM is impractical to use for large-scale data.

We also performed a face image reconstruction experiment using the proposed methods and the other methods in the presence of occlusions and missing data. Occlusion blocks were generated as described in Section V-B in 50% randomly selected images. To generate missing blocks, 50% of images were randomly selected again, and a randomly located square block, whose side length varied from 30 to 60 pixels, was considered as missing in each image. The values of the block elements were set to zero. The number of projection vectors was set to 100. The average reconstruction error E_1 and execution time for various methods are shown in Table VI. In the table, Wl_1-ARG$_D$ shows good performance in both of the reconstruction error and execution time compared with the other methods. Although Regl_1-ALM gives the comparable reconstruction error with the proposed methods, its computation time is longer than the proposed methods. Fig. 4 shows the reconstructed face images in the presence of occlusions and missing data. We do not see much difference between the reconstructed images of the proposed methods and Regl_1-ALM in this figure.

D. Nonrigid Motion Estimation

Nonrigid motion estimation [28] with outliers and missing data from image sequences can be considered as a factorization problem. In this problem, l_1-norm-based factorization can be applied to restore 2-D tracks contaminated by outliers and missing data. In this experiment, we used the

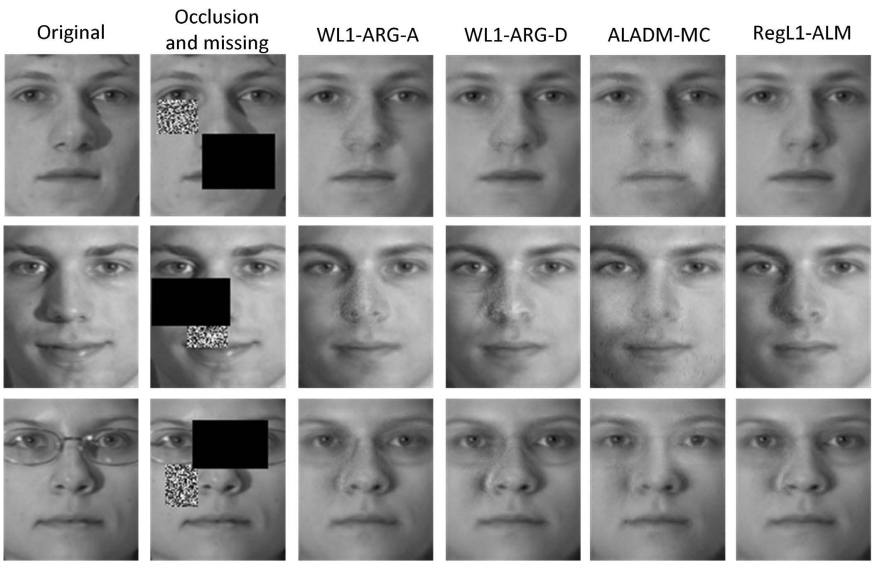


Fig. 4. Face images with occlusions and missing blocks, and their reconstructed faces.

TABLE VII
RECONSTRUCTION RESULTS FOR GIRAFFE SEQUENCE IN THE PRESENCE OF ADDITIONAL OUTLIERS

	10% outliers		20% outliers	
Algorithm	Error	Time (sec)	Error	Time (sec)
Wl_1-ARG$_A$	2.910	3.623	3.006	1.589
Wl_1-ARG$_D$	3.224	1.217	3.950	0.895
Wl_1-ARG$_{A+D}$	2.847	4.051	2.979	1.754
Regl_1-ALM	3.792	0.810	3.939	0.820
ALADM-MC	9.835	0.017	21.908	0.013

well-known giraffe sequence[2] consisting of 166 tracked points and 120 frames. The data size is 240×166 and 30.24% of entries are missing. In this section, we also present another algorithm, Wl_1-ARG$_{A+D}$, which is Wl_1-ARG$_D$ using the result of Wl_1-ARG$_A$ as an initial value. The goal of using Wl_1-ARG$_{A+D}$ is to verify the superiority of Wl_1-ARG$_D$ compared with Wl_1-ARG$_A$ by showing that Wl_1-ARG$_D$ can improve the quality of the solution beyond what is possible by Wl_1-ARG$_A$.

To demonstrate the robustness of the proposed methods, we replaced 10% of the points in a frame by outliers in the range $[-1000, 2000]$, whereas the data points are in the range $[127, 523]$. In another experiment, we constructed the data by replacing 20% of the points in a frame by outliers. The number of shape bases was set to 2, which gave a matrix of rank $6 = 2 \times 3$ (for x, y, and z coordinates). We compared the weighted versions of the proposed algorithms with ALADM-MC and Regl_1-ALM. We set the stopping condition ρ to 10^{-6} and β in (49) to 10^{-1}. The result of reconstruction error[3] for the observation data can be observed in Table VII. As shown in the table, Wl_1-ARG$_A$ gives a better performance than Wl_1-ARG$_D$ but poor than Wl_1-ARG$_{A+D}$ in this problem. We suspect that Wl_1-ARG$_D$ is more sensitive to the initial value and can be trapped in a local minimum for a complex problem. Thus, Wl_1-ARG$_A$ can sometimes find a better solution than Wl_1-ARG$_D$. However, when we apply Wl_1-ARG$_D$ with a good initial value, such as a solution found by Wl_1-ARG$_A$, we can improve the quality of the solution further. It suggests that the combination Wl_1-ARG$_{A+D}$ can be a good approach for many complex problems. Although ALADM-MC takes shorter execution time than the other methods, it gives poor reconstruction results. Regl_1-ALM gives the competitive results compared with Wl_1-ARG$_A$ with respect to to the error and execution time in this experiment.

We also performed the nonrigid motion estimation problem using the shark sequence [28] which consists of 91 tracked points for each nonrigid shark shape in 240 frames. In this data, we examine how robust the proposed methods are for various missing ratios in the presence of outliers. We replaced 10% of the points in each frame by outliers in the range $[-1000, 1000]$, whereas the data points were located in the range $[-105, 105]$. We set from 10% to 70% of tracked points as missing in each frame. The number of shape basis for each coordinate was set to two, thus it can be formulated as a rank-6 approximation problem.

The average performance for the various methods are shown in Table VIII. Similar to Table VII, Wl_1-ARG$_A$ gives better reconstruction results than Wl_1-ARG$_D$ for this problem but performs worse than Wl_1-ARG$_{A+D}$ due to the approximated nature of Wl_1-ARG$_A$. Although Regl_1-ALM gives an excellent reconstruction error when 10% and 30% of data were missing,

[2]Available at http://www.robots.ox.ac.uk/∼abm/

[3]Reconstruction error is calculated as stated at http://www.robots.ox.ac.uk/∼abm/

TABLE VIII
AVERAGE PERFORMANCE FOR SHARK SEQUENCE WITH 10% OUTLIERS AND MISSING DATA

	missing 10%		missing 30%		missing 50%		missing 70%	
Algorithm	Error	Time (sec)	Error	Time (sec)	Error	Time (sec)	Error	Time (sec)
Wl_1-ARG$_A$	0.069	0.562	0.106	0.819	0.460	0.660	1.767	1.590
Wl_1-ARG$_D$	0.266	0.078	0.366	0.217	0.929	0.233	3.101	0.895
Wl_1-ARG$_{A+D}$	0.063	0.615	0.087	0.895	0.443	0.744	1.676	1.889
$Regl_1$-ALM	0.032	0.805	0.039	0.815	2.739	0.872	24.806	0.364
ALADM-MC	0.402	0.025	0.942	0.023	7.449	0.206	10.015	0.029

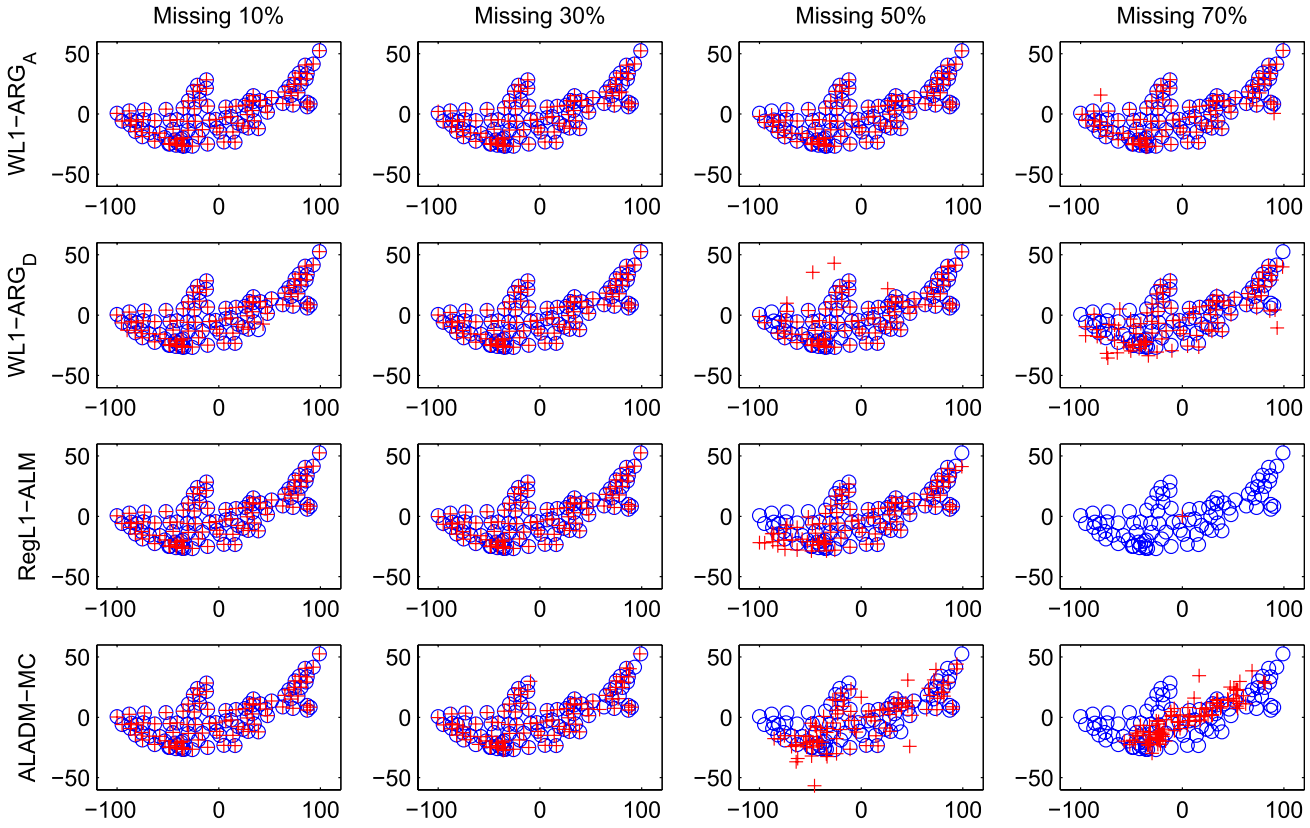


Fig. 5. Nonrigid shape estimation from shark image sequences for various missing ratios.

but its performance gets worse as the missing data increases. The reconstruction results for a few selected frames are shown in Fig. 5.

VI. CONCLUSION

In this paper, we have proposed two novel methods, l_1-ARG$_A$ and l_1-ARG$_D$, for low-rank matrix approximation problems, using the l_1-norm-based alternating rectified gradient method. We also have shown how to apply the proposed methods when some of the data are missing. For l_1-ARG$_D$, an alternating rectified gradient method based on dual formulation, we have proved the convergence of the algorithm to the subspace-wise local minimum using the global convergence theorem. Although it is hard for l_1-ARG$_A$ to guarantee convergence to a local minimum, it works well in practical problems. The experimental results have shown that the proposed methods provide an excellent reconstruction performance and a shorter execution time compared with the other methods except for a few cases. Even in such cases, the performance difference is insignificant. We have shown the superiority of the proposed methods for large-scale problems.

REFERENCES

[1] D. M. Hawkins, L. Liu, and S. S. Young, "Robust singular value decomposition," Nat. Inst. Statist. Sci., Triangle Park, NC, USA, Tech. Rep. 122, 2001.
[2] Q. Ke and T. Kanade, "Robust subspace computation using L_1 norm," School Comput. Sci., Carnegie Mellon Univ., Pittsburgh, PA, USA, Tech. Rep. CMU-CS-03-172, Aug. 2003.
[3] Q. Ke and T. Kanade, "Robust L_1 norm factorization in the presence of outliers and missing data by alternative convex programming," in *Proc. IEEE Conf. CVPR*, Jun. 2005, pp. 739–746.
[4] C. Ding, D. Zhou, X. He, and H. Zha, "R_1-PCA: Rotational invariant l_1-norm principal component analysis for robust subspace factorization," in *Proc. IEEE ICML*, Jun. 2006, pp. 281–288.
[5] N. Kwak, "Principal component analysis based on L_1-norm maximization," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 3, no. 9, pp. 1672–1680, Sep. 2008.

[6] J. Brooks, J. Dula, and E. Boone, "A pure L_1-norm principal component analysis," *Comput. Statist. Data Anal.*, vol. 61, pp. 83–98, May 2013.

[7] X. Ding, L. He, and L. Carin, "Bayesian robust principal component analysis," *IEEE Trans. Image Process.*, vol. 20, no. 12, pp. 3419–3430, Dec. 2011.

[8] Y. Shen, Z. Wen, and Y. Zhang, "Augmented Lagrangian alternating direction method for matrix separation based on low-rank factorization," *Optim. Methods Softw.*, vol. 29, no. 2, pp. 239–263, 2014.

[9] A. Eriksson and A. van den Hengel, "Efficient computation of robust weighted low-rank matrix approximations using the L_1 norm," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 9, pp. 1681–1690, Sep. 2012.

[10] Y. Zheng, G. Liu, S. Sugimoto, S. Yan, and M. Okutomi, "Practical low-rank matrix approximation under robust L_1-norm," in *Proc. IEEE Conf. CVPR*, Jun. 2012, pp. 1410–1417.

[11] I. T. Jolliffe, *Principal Component Analysis*. New York, NY, USA: Springer-Verlag, 1986.

[12] Z. Lin, M. Chen, L. Wu, and Y. Ma, "The augmented Lagrange multiplier method for exact recovery of corrupted low-rank matrices," Univ. Illinois at Urbana-Champaign, Champaign, IL, USA, Tech. Rep. UILU-ENG-09-2215, 2009.

[13] S. Jot and J. P. Brooks, "pcaL_1: Three L_1-norm PCA methods," M.S. thesis, Dept. Math. Sci., Virginia Commonwealth Univ., Richmond, VA, USA, 2012.

[14] J. Gao, "Robust L_1 principal component analysis and its Bayesian variational inference," *Neural Comput.*, vol. 20, no. 2, pp. 555–572, 2008.

[15] K. Mitra, S. Sheorey, and R. Chellappa, "Large-scale matrix factorization with missing data under additional constraints," in *Proc. Adv. NIPS*, vol. 23. 2010, pp. 1642–1650.

[16] K.-C. Toh and S. Yun, "An accelerated proximal gradient algorithm for nuclear norm regularized least squares problems," *J. Optim.*, vol. 6, no. 3, pp. 615–640, 2010.

[17] X. Yuan and J. Yang, "Sparse and low-rank matrix decomposition via alternating direction method," *Pacific J. Optim.*, vol. 9, no. 1, pp. 167–180, 2013.

[18] E. J. Candes, X. Li, Y. Ma, and J. Wright, "Robust principal component analysis?" *J. ACM*, vol. 58, no. 3, pp. 11:1–11:37, 2011.

[19] J. Liu, P. Musialski, P. Wonka, and J. Ye, "Tensor completion for estimating missing values in visual data," in *Proc. IEEE 12th ICCV*, Oct. 2009, pp. 2114–2121.

[20] Y. Li, J. Yan, Y. Zhou, and J. Yang, "Optimum subspace learning and error correction for tensors," in *Proc. ECCV*, 2010, pp. 790–803.

[21] C. Gurwitz, "Weighted median algorithms for L_1 approximation," *BIT*, vol. 30, no. 2, pp. 301–310, 1990.

[22] I. Csiszar and G. Tusnady, "Information geometry and alternating minimization procedures," *Statist. Decisions, Supplement Issue*, vol. 1, no. 1, pp. 205–237, 1984.

[23] C. A. R. Hoare, "Algorithm 65: Find," *Commun. ACM*, vol. 4, no. 7, pp. 321–322, Jul. 1961.

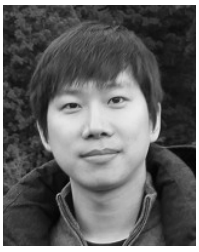
[24] A. Rauh and G. R. Arce, "A fast weighted median algorithm based on quickselect," in *Proc. IEEE Conf. Image Process.*, Sep. 2010, pp. 105–108.

[25] W. I. Zangwill, *Nonlinear Programming: A Unified Approach*. Englewood Cliffs, NJ, USA: Prentice-Hall, 1969.

[26] D. G. Luenberger, *Linear and Nonlinear Programming*. New York, NY, USA: Springer-Verlag, 2010.

[27] R. Gross, I. Matthews, J. Cohn, T. Kanade, and S. Baker, "Multi-PIE," in *Proc. IEEE Conf. Autom. Face Gesture Recognit.*, Sep. 2008, pp. 1–8.

[28] L. Torresani, A. Hertzmann, and C. Bregler, "Learning non-rigid 3D shape from 2D motion," in *Proc. NIPS*, 2003, pp. 1555–1562.



Eunwoo Kim (S'12) received the B.S. degree in electrical and electronics engineering from Chung-Ang University, Seoul, Korea, and the M.S. degree in electrical engineering and computer sciences from Seoul National University, Seoul, in 2011 and 2013, respectively, where he is currently pursuing the Ph.D. degree with the Department of Electrical and Computer Engineering.

His current research interests include pattern recognition, computer vision, machine learning, image processing, and their applications.

Minsik Lee (S'11–M'13) received the B.S. and Ph.D. degrees from the School of Electrical Engineering and Computer Science, Seoul National University, Seoul, Korea, in 2006 and 2012, respectively.

He is currently a BK21 Assistant Professor in the Graduate School of Convergence Science and Technology, Seoul National University, Suwon, Korea. His current research interests include computer vision, pattern analysis, machine learning, image processing, and their applications.

Chong-Ho Choi (M'09) received the B.S. degree from Seoul National University, Seoul, Korea, and the M.S. and Ph.D. degrees from the University of Florida, Gainesville, FL, USA, in 1970, 1975, and 1978, respectively.

He was a Senior Researcher at the Korea Institute of Science and Technology, Seoul, Korea, from 1978 to 1980, and served as a Professor in the School of Electrical and Computer Engineering, Seoul National University, from 1980 to 2013, where he is an Emeritus Professor. His current research interests include control theory, wireless networks, pattern recognition, and their applications.

Nojun Kwak (M'00) was born in Seoul, Korea, in 1974. He received the B.S., M.S., and Ph.D. degrees from the School of Electrical Engineering and Computer Science, Seoul National University, Seoul, Korea, in 1997, 1999, and 2003, respectively.

He was with Samsung Electronics, Suwon, Korea, from 2003 to 2006. In 2006, he joined Seoul National University as a BK21 Assistant Professor. From 2007 to 2013, he was a Faculty Member of the Department of Electrical and Computer Engineering, Ajou University, Suwon, Korea. Since 2013, he has been with the Graduate School of Convergence Science and Technology, Seoul National University, Suwon, Korea, where he is currently an Associate Professor. His current research interests include pattern recognition, machine learning, computer vision, data mining, image processing, and their applications.

Songhwai Oh (S'04–M'07) received the B.S. (Hons.), M.S., and Ph.D. degrees in electrical engineering and computer sciences from the University of California, Berkeley, CA, USA, in 1995, 2003, and 2006, respectively.

He is currently an Associate Professor in the Department of Electrical and Computer Engineering, Seoul National University, Seoul, Korea. Before the Ph.D. studies, he was a Senior Software Engineer at Synopsys, Inc., Mountain View, CA, USA, and a Microprocessor Design Engineer at Intel Corporation, Santa Clara, CA, USA. In 2007, he was a Post-Doctoral Researcher at the Department of Electrical Engineering and Computer Sciences, University of California, Berkeley, CA, USA. From 2007 to 2009, he was an Assistant Professor of electrical engineering and computer science in the School of Engineering, University of California, Merced, CA, USA, from 2007 to 2009. His current research interests include cyber-physical systems, robotics, computer vision, and machine learning.