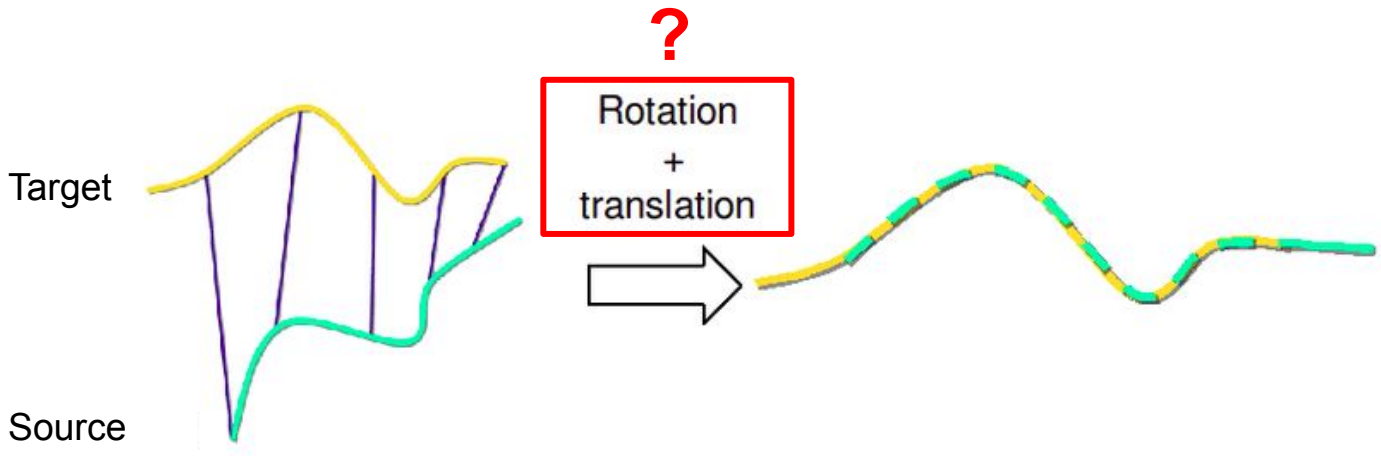
AA-ICP

Iterative Closest Point with Anderson Acceleration

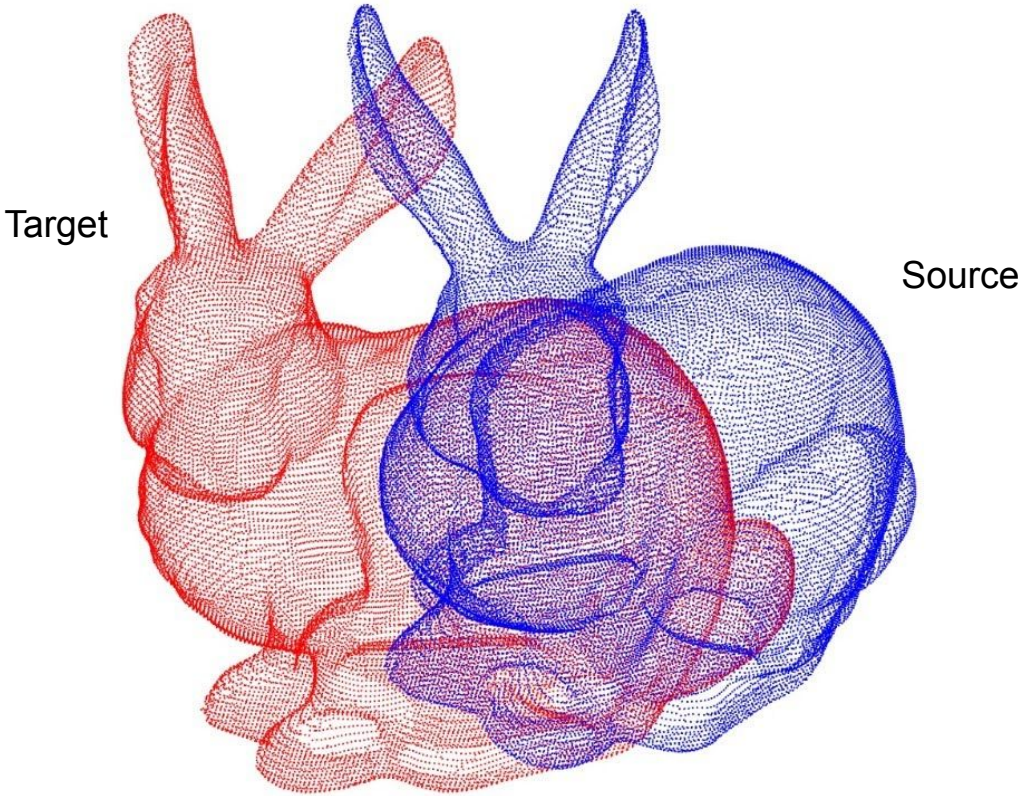
Team members: Dmitriy Derbyshev, Georgiy Kozhevnikov

Problem

The goal is to find a transformation which will translate source point cloud to target point cloud



3D example



ICP (iterative closest point) algorithm

Iterative algorithm that converges to transformation matrix that transforms source cloud point to be as close to target cloud point.

In the most simple case, algorithm is as follows:

Given source and target point clouds S and S_t:

1. Transform S_t using initial transformation estimation T_0
2. For every point x in S_t find the closest point y in S
3. Find such transformation T_i which minimizes mean distance between these pairs (MSE)
4. Apply T_i to S_t
5. If change in the error is less than a given threshold - terminate; otherwise - go to step 2

ICP (iterative closest point) algorithm

It can be shown that ICP algorithm is essentially boils down to a fixed point problem:

$u_k = G(u_{k-1})$, where

$u_k = T_k T_{k-1} \ldots T_0$ - estimation of the desired transformation;

G - Iteration of ICP algorithm

It is usually solved by **Picard iteration procedure:**

$u_{k+1} = G(u_k)$

Anderson acceleration

Another algorithm which solves the fixed point problem using history of `m+1` latest iterations.

In case of `m=0` it is equivalent to Picard iterative procedure.

In general, Anderson acceleration is theoretically and practically superior to Picard in many cases

Algorithm 1: Anderson acceleration

Data: initial guess u^0, contraction mapping G,
maximum iterations limit n_{max}
Result: fixed point u
$g^0 = G(u^0)$;
$f^0 = g^0 - u^0$;
$u^1 = g^0$;
for n *in* $1..n_{max}$ **do**
  $f^n = G(u^n) - u^n$;
  Find $\alpha \in \mathbf{R}^{n+1}$ which minimizes $\| \sum_{j=0}^{n} \alpha_j f^j \|_2$
  subject to $\sum_j \alpha_j = 1$;
  $u^{n+1} = \sum_{j=0}^{n} \alpha_j G(u^j)$;
  if *convergence criteria is true* **then**
    break;
  end
end
return u^n;

AA-ICP (Anderson accelerated iterative closest point) algorithm

Given source and target point clouds S and S_t:

1. Transform S_t using initial transformation estimation T_0
2. For every point x in S_t find the closest point y in S
3. Find such transformation T_i which minimizes mean distance between these pairs (MSE)
4. **Change T_i using history of previous transformations and residues of these transformations**
5. Apply T_i to S_t
6. If change in the error is less than a given threshold - terminate; otherwise - go to step 2

AA-ICP

Idea of the approach is to improve convergence of ICP using Anderson acceleration.

Algorithm 2: AA-ICP

Data: initial guess u^0, history length limit m, alpha limit α_l, maximum iterations limit n_{max}
Result: convergence point u^{n+1}

$h = 0$; // history cut-off cursor
$g^0 = G(u^0)$;
$f^0 = g^0 - u^0$;
$u^1 = g^0$;
for n *in* $1..n_{max}$ **do**
  $g^n = G(u^n)$;
  $f^n = g^n - u^n$;
  if *ICP error is too big* **then**
    $h = n$; // "resetting" history
    $u^{n+1} = g^{n-1}$;
    continue;
  end
  $u^{n+1} = g^n$;
  for i *in* $1..min(m, n-h)$ **do**
    $\alpha_{1,\dots,n} = \min \|f_0 + \sum_{j=1}^{n} \alpha_j (f_j - f_0)\|$;
    $\alpha_0 = 1 - \sum_{j=1}^{n} \alpha_j$;
    if $(-\alpha_l \le \alpha_j \le \alpha_l, \forall j)$ *and* $(\alpha_0 > 0)$ **then**
      $u^{n+1} = \sum_{j=0}^{i} \alpha_j g^{n-j}$;
    else
      break;
    end
  end
  if *convergence criteria is true* **then**
    break;
  end
end
return u^{n+1};

AA-ICP

However in real world data G is usually non-contractive, which causes instability of Anderson algorithm.

That is a reason to use some heuristics that allow to avoid these instabilities.

Algorithm 2: AA-ICP

Data: initial guess u^0, history length limit m, alpha limit α_l, maximum iterations limit n_{max}
Result: convergence point u^{n+1}

$h = 0$; // history cut-off cursor
$g^0 = G(u^0)$;
$f^0 = g^0 - u^0$;
$u^1 = g^0$;
for n *in* $1..n_{max}$ **do**
  $g^n = G(u^n)$;
  $f^n = g^n - u^n$;
  if *ICP error is too big* **then**
    $h = n$; // "resetting" history
    $u^{n+1} = g^{n-1}$;
    continue;
  end
  $u^{n+1} = g^n$;
  for i *in* $1..min(m, n-h)$ **do**
    $\alpha_{1,\dots,n} = \min \|f_0 + \sum_{j=1}^{n} \alpha_j (f_j - f_0)\|$;
    $\alpha_0 = 1 - \sum_{j=1}^{n} \alpha_j$;
    if $(-\alpha_l \le \alpha_j \le \alpha_l, \forall j)$ *and* $(\alpha_0 > 0)$ **then**
      $u^{n+1} = \sum_{j=0}^{i} \alpha_j g^{n-j}$;
    else
      break;
    end
  end
  if *convergence criteria is true* **then**
    break;
  end
end
return u^{n+1};

AA-ICP

There are 3 heuristics:

1. If ICP error is too big we reset the history and do single Picard iteration
2. If `alpha_0` coefficient is less than 0, we do not apply Anderson acceleration
3. If `alpha_i` are not inside the given limits, we do not apply Anderson acceleration

Algorithm 2: AA-ICP

Data: initial guess u^0, history length limit m, alpha limit α_l, maximum iterations limit n_{max}
Result: convergence point u^{n+1}

```
h = 0;                          // history cut-off cursor
g^0 = G(u^0);
f^0 = g^0 − u^0;
u^1 = g^0;
for n in 1..n_max do
    g^n = G(u^n);
    f^n = g^n − u^n;
    if ICP error is too big then                                   1
        h = n;                  // "resetting" history
        u^{n+1} = g^{n−1};
        continue;
    end
    u^{n+1} = g^n;
    for i in 1..min(m, n − h) do
        α_{1,...,n} = min ‖f_0 + Σ_{j=1}^{n} α_j (f_j − f_0)‖;
        α_0 = 1 − Σ_{j=1}^{n} α_j;
        if (−α_l ≤ α_j ≤ α_l, ∀j) and (α_0 > 0) then              2, 3
            u^{n+1} = Σ_{j=0}^{i} α_j g^{n−j};
        else
            break;
        end
    end
    if convergence criteria is true then
        break;
    end
end
return u^{n+1};
```

Practical notes

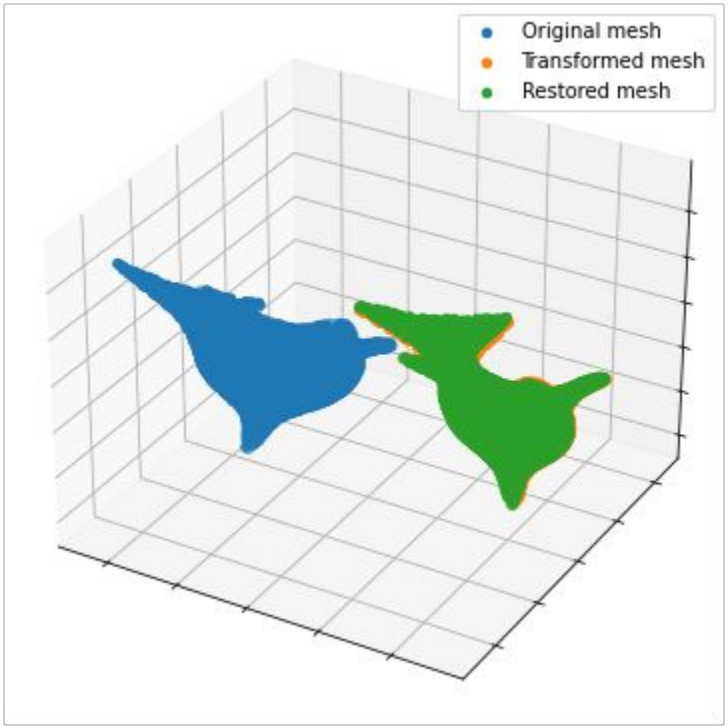
We consider only roto-translations RT which are usually represented using rotation matrix R and translation vector T.

In practice, we need a way to represent the transformation as a vector to apply these algorithms.

For experiments we try 2 approaches:

1. Represent transformation as a concatenation of T and flattened R:
   ```
   RT -> (R_00, R_01, …, R_22, T_0, T_1, T_2)
   ```
2. Translate R into euler angles and concatenate with T:
   ```
   RT -> (fi, psi, hi,T_0,T_1,T_2)
   ```

Experiments



Results of ICP algorithm on a Dolphin mesh

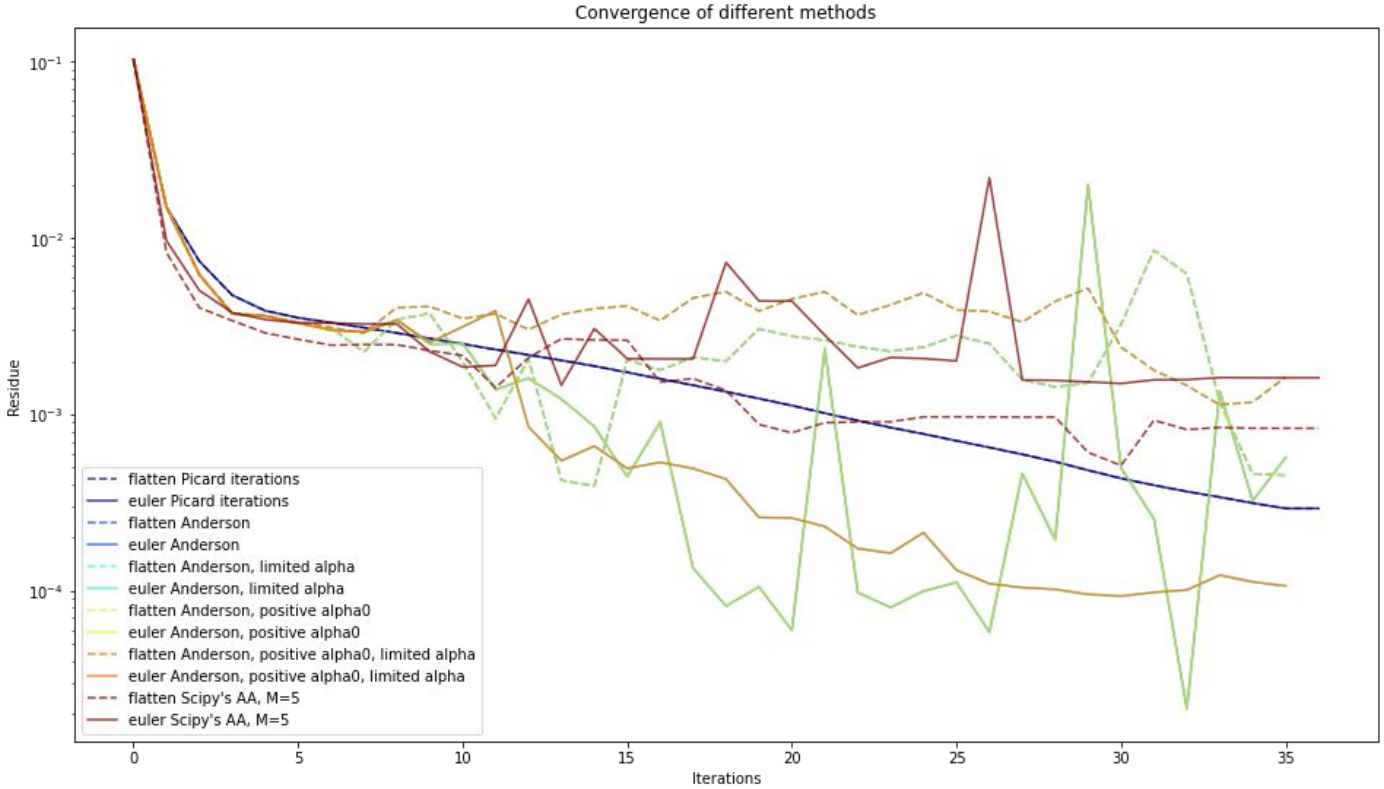
Experiments

We studied behaviour of the algorithm with and without heuristics in different conditions.

Factors we examined:

1) Combinations of heuristics
2) Different representations of transformation matrix
3) `alpha_i` constraints
4) Memory size

Experiments 1-2

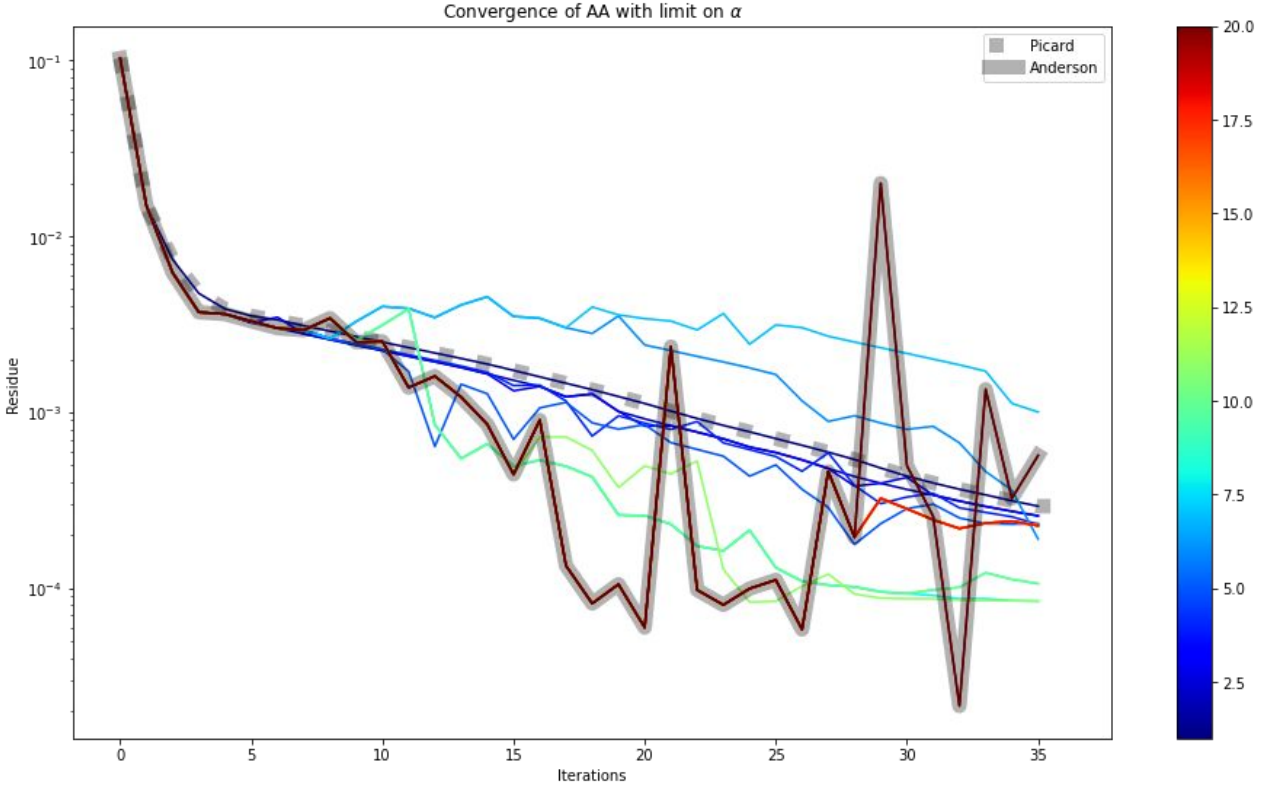


Experiments 1-2

We obtained several conclusions:

- Rotational matrix should be transformed to Euler angles, not simply flattened
- Limitations on `alpha_i` are crucial to avoid incorrect jumps of `u`
- In case of limited `alpha_i`, additional limitation `alpha_0 > 0` does not affect convergence

Experiments 3

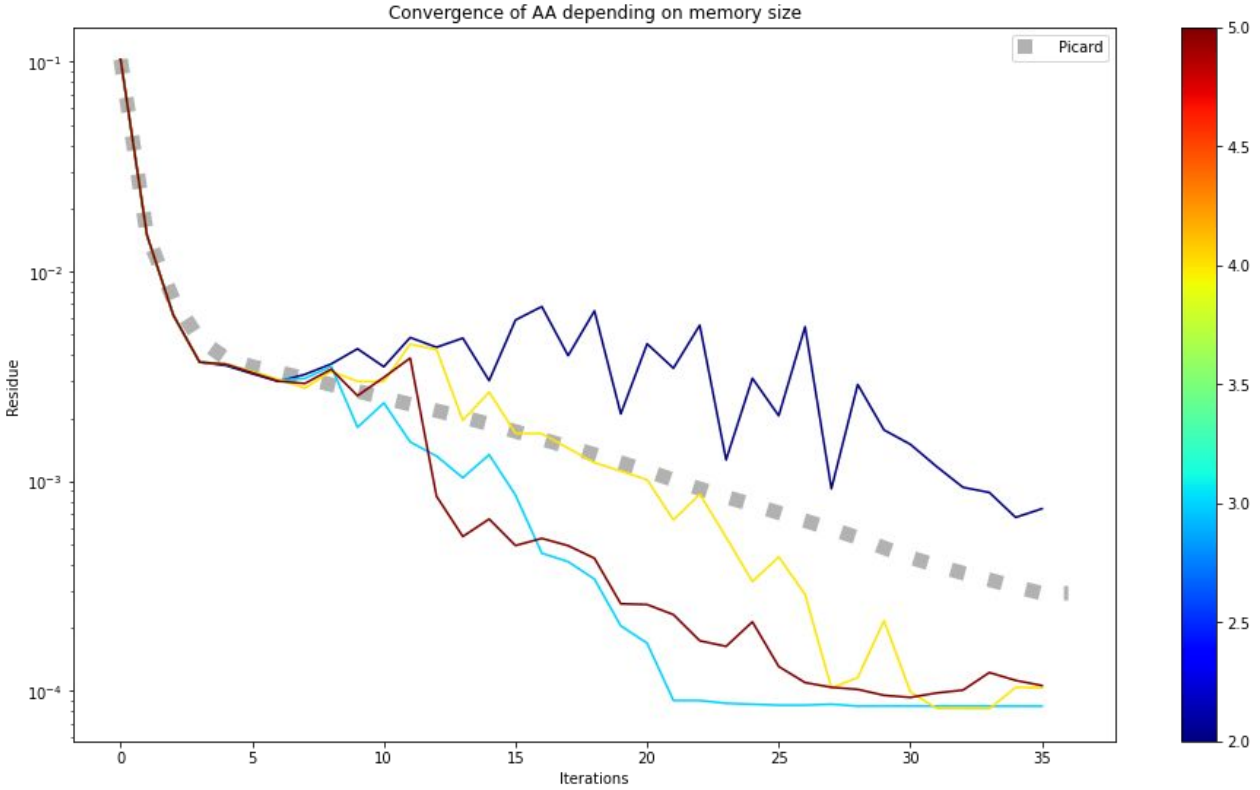


Experiments 3

Conclusions:

- Too strict limitations on `alpha_i` force Picard iterations
- Too loose limitations does not affect convergence of ICP

Experiment 4



Experiment 4

Conclusions:

- With history length of 1 Anderson performs just as Picard iterations
- The best results in terms of convergence achieved by storing as much history as possible, up until the size of linearized transformation matrix
- Storing more history is not practical, since optimization problem (least squares) inside Anderson algorithm becomes over-determined