解释table/star/.em/.tbl…

黄振

.em是Motive List(motl): 如何得到每个particle的朝向

motl: 20行，列数就是particle数量(用coos_keep行数获得)

coos_keep: 等值线上采样点particle的坐标

遍历motl的每一列(每一个particle)

X’ Y’ Z’

-round(500*n(i,1)) 表示将法向量 n(i,:) 在 X 方向上延伸 500 个单位长度，并取整。获得位于500距离处法向量反向延伸线上的一个固定距离的点的坐标X’

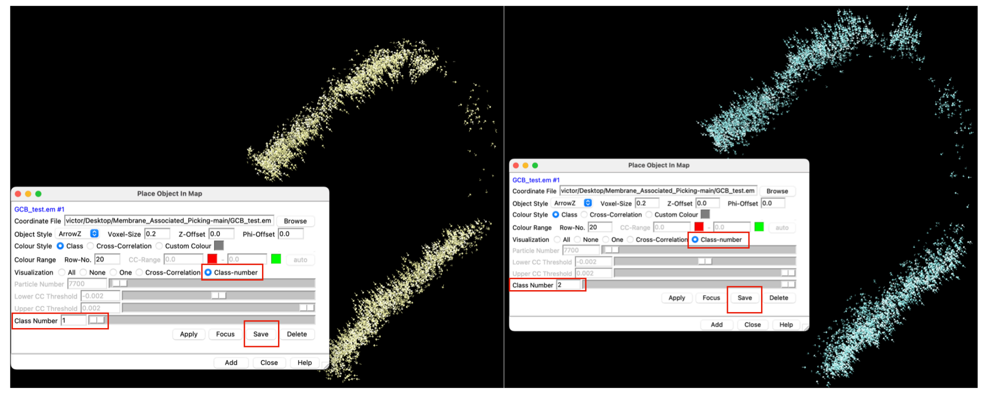
dist_p_to_cent表示particle到重心的距离
dist_ison_to_cent表示(X’,Y’,Z’)到重心的距离

cent记录三维体积数据在x、y和z方向上的重心坐标。

```
%write motivelist with coordinates and run normalvec. motl will need to be modified to include tomogram number particle numbers etc.
motl=zeros(20,size(coos_keep,1));
motl(8:10,:)=transpose(coos_keep);
motl_n=[];

for i=1:size(motl,2)
        motl_i=normalvec(motl(:,i),[coos_keep(i,1)-round(500*n(i,1)),coos_keep(i,2)-round(500*n(i,2)),coos_keep(i,3)-round(500*n(i,3))]);
        dist_p_to_cent=(motl(8,i)-cent(1)).^2+(motl(9,i)-cent(2)).^2+(motl(10,i)-cent(3)).^2;
                dist_ison_to_cent=((coos_keep(i,1)-round(500*n(i,1)))-cent(1)).^2+((coos_keep(i,2)-round(500*n(i,2)))-cent(2)).^2+((coo
        if dist_p_to_cent>dist_ison_to_cent
                motl_i(20,1)=1;
                end
        if dist_p_to_cent<dist_ison_to_cent
                motl_i(20,1)=2;
                end

        motl_n=cat(2,motl_n,motl_i);
end
motl_n(8:10,:)=transpose([coos_keep(:,1)-round(500*n(:,1)),coos_keep(:,2)-round(500*n(:,2)),coos_keep(:,3)-round(500*n(:,3))]);

%write out your motivelist
motlname=[surface_filename '.em'];
tom_emwrite_mod(motlname,motl_n);
end

if size(surf.vertices)==[0,0]
motl_n=[];
motlname=[surface_filename '.em'];
tom_emwrite(motlname,motl_n);
end
```

如果X’远离重心，class为1

如果X’离重心更近，class为2



记录X’，Y’，Z’的坐标

.em是Motive List(motl)

1. .em的列数就是particles的数量；
2. .em的行包含各种参数；

```
function motl = normalvec(motl, cent)
% NORMALVEC determines Euler angles of particles on a sphere
%
%   motl = normalvec(motl, cent)
%
%   assume particles are on sphere with center = cent. Euler angles psi and
%   theta are calculated and stored in motl.
%
% PARAMETERS
%  INPUT
%   motl          : motivelist
%   cent          : center (3D vector)
%
%  OUTPUT
%   MOTL            motivelist with new angles
```

.em是Motive List(motl): 如何通过向量计算欧拉角

1. .em的列数就是particles的数量；
2. .em的行包含各种参数；

```
disp(['x= ' num2str(cent(1)) 'y= ' num2str(cent(2)) 'z= ' num2str(cent(3))])
npart = size(motl,2);
tmp(1:npart) = cent(1);
x = motl(8,:)-tmp;
tmp(1:npart) = cent(2);
y = motl(9,:)-tmp;
tmp(1:npart) = cent(3);
z = motl(10,:)-tmp;
tmp(:) = 90;
theta= 180/pi*atan2(sqrt(x.^2+y.^2),z); %theta
psi  = 90+180/pi*atan2(y,x); %psi
motl(18,:)=psi;
motl(19,:)=theta;
```

em列数就是particles数量

获得位于500距离处法向量反向延伸线上的一个固定距离的点的坐标X’, 记录在cent(1)中

x=△x = X-X’

空间向量x,y,z
与法向量方向相同

这个是怎么计算的呢？psi是代表绕x轴、theta是代表绕z轴吗？
初始朝向又是哪个向量呢？具体是怎么旋转的呢？为什么psi要加90度呢？…

MAP用的就是av3的计算方法得到.em

normalvec.m

Pull requests Check out

Collection of matlab functions for processing (cryo) electron tomographic data. For example, Template matching, subtomogram averaging and classification are supported.

Source master 8e290c2 Full commit

av3 / utils / normalvec.m

Edit

```
%   assume particles are on sphere with center = cent. Euler angles psi and
%   theta are calculated and stored in motl.
%
% PARAMETERS
%  INPUT
%   motl        : motivelist
%   cent        : center (3D vector)
%
%  OUTPUT
%   MOTL           motivelist with new angles
%
% last change 03/31/05 FF - docu updated
%
%   Copyright (c) 2005-2010
%   Max-Planck-Institute for Biochemistry
%   Dept. Molecular Structural Biology
%   82152 Martinsried, Germany
%   http://www.biochem.mpg.de/foerster
%
disp(['x= ' num2str(cent(1)) 'y= ' num2str(cent(2)) 'z= ' num2str(cent(3))])
npart = size(motl,2);
tmp(1:npart) = cent(1);
x = motl(8,:)-tmp;
tmp(1:npart) = cent(2);
y = motl(9,:)-tmp;
tmp(1:npart) = cent(3);
z = motl(10,:)-tmp;
tmp(:) = 90;
theta= 180/pi*atan2(sqrt(x.^2+y.^2),z); %theta
psi  = 90+180/pi*atan2(y,x); %psi
motl(18,:)=psi;
motl(19,:)=theta;
```

只计算了theta和psi，phi取0

AV3的motivelist (.em) to dynamo的.tbl

```
list=dir('TS_*_object_*.em'); % This line tells MatLab to look inside all files with this r

%%%END OF USER INPUT SECTION

list_names={list.name}; % This extracts the file names

for i = 1:length(list_names) % This loops through all the files

    tomon=list_names{1,i};
    tomon=char(extractBetween(tomon,'TS_','_object')); % THis extracts the tomogram number
    tuben=list_names{1,i};
    tuben=char(extractBetween(tuben,'_object_','.em')); % THis extracts the object number

    motl=dread(list_names{1,i}); % Reading motl file
    table=dynamo__motl2table(motl); % Converting to table
    table(:,20)=str2num(tomon); %Entering tomogram number into table
    table(:,21)=str2num(tuben); %Entering object number into table
    if pick_particle=='y'
    table(:,24:26)=(table(:,24:26)./pixel_size);
    end
    dwrite(table,['TS_' tomon '_object_' tuben '.tbl']); %Saving as .tbl
end
```

AV3的motivelist (.em) to dynamo的.tbl

```
% full conversion of AV3-style motl to Dynamo-style table
%
% INPUT
%     motl AV3             as command line variable
%     wedgelist            as command line variable
%
% OUTPUT
%     table:               as command line variable
%


% Note: this function uses the following AV3 convention:
```

+5 dynamo2motl.m motl2dynamo.m dynamo__motl2table.m +

```
%        5          : running number of tomogram - used for wedgelist
%        6          : index of feature in tomogram (optional)
%        8          : x-coordinate in full tomogram
%        9          : y-coordinate in full tomogram
%        10         : z-coordinate in full tomogram
%        11         : x-shift in subvolume - AFTER rotation of template
%        12         : y-shift in subvolume - AFTER rotation of template
%        13         : z-shift in subvolume - AFTER rotation of template
%      ( 14         : x-shift in subvolume - BEFORE rotation of template
%      ( 15         : y-shift in subvolume - BEFORE rotation of template
%      ( 16         : z-shift in subvolume - BEFORE rotation of template
%        17         : Phi (in deg)
%        18         : Psi
%        19         : Theta
%        20         : class no
```

只计算了theta和psi，phi取0

.em to .tbl: 真正的顺序是psi, theta, phi

在dynamo内置函数: dynamo__motl2table.m中：

```
% reads from motl
identities=motl(4,:);
phi=motl(17,:);
psi=motl(18,:);
theta=motl(19,:);

xshift=motl(11,:);
yshift=motl(12,:);
zshift=motl(13,:);
```

```
% NOTE:
% convention previous to dynamo 0.8
% narot=phi';
% tilt=theta';
% tdrot=psi';

% convention after dynamo 0.8

tdrot=-psi';
tilt=-theta';
narot=-phi';

% NOTE
% A sanity check for the angular convention:
% [tdrot,tilt,narot]=dynamo_angles_random;dynamo_slices({dynamo_rot(a,[tdrot,tilt,narot]),rot(a,-[
% % rotation syntax in TOM: phi,psi,theta
%
% It is apparently working correctly.
```

T表示逆时针旋转某角，那么-T'表示顺时针旋转该角

相当于旋转tdrot，再转tilt，narot取0（因为phi取了0）

tdrot, tilt, narot才是真的zxz

dynamotable/constants.py

```
    'aligned_value': 2,
    'averaged_value': 3,
    'dx': 4,
    'dy': 5,
    'dz': 6,
    'tdrot': 7,
    'tilt': 8,
    'narot': 9,
```

分别代表三个欧拉角zxz

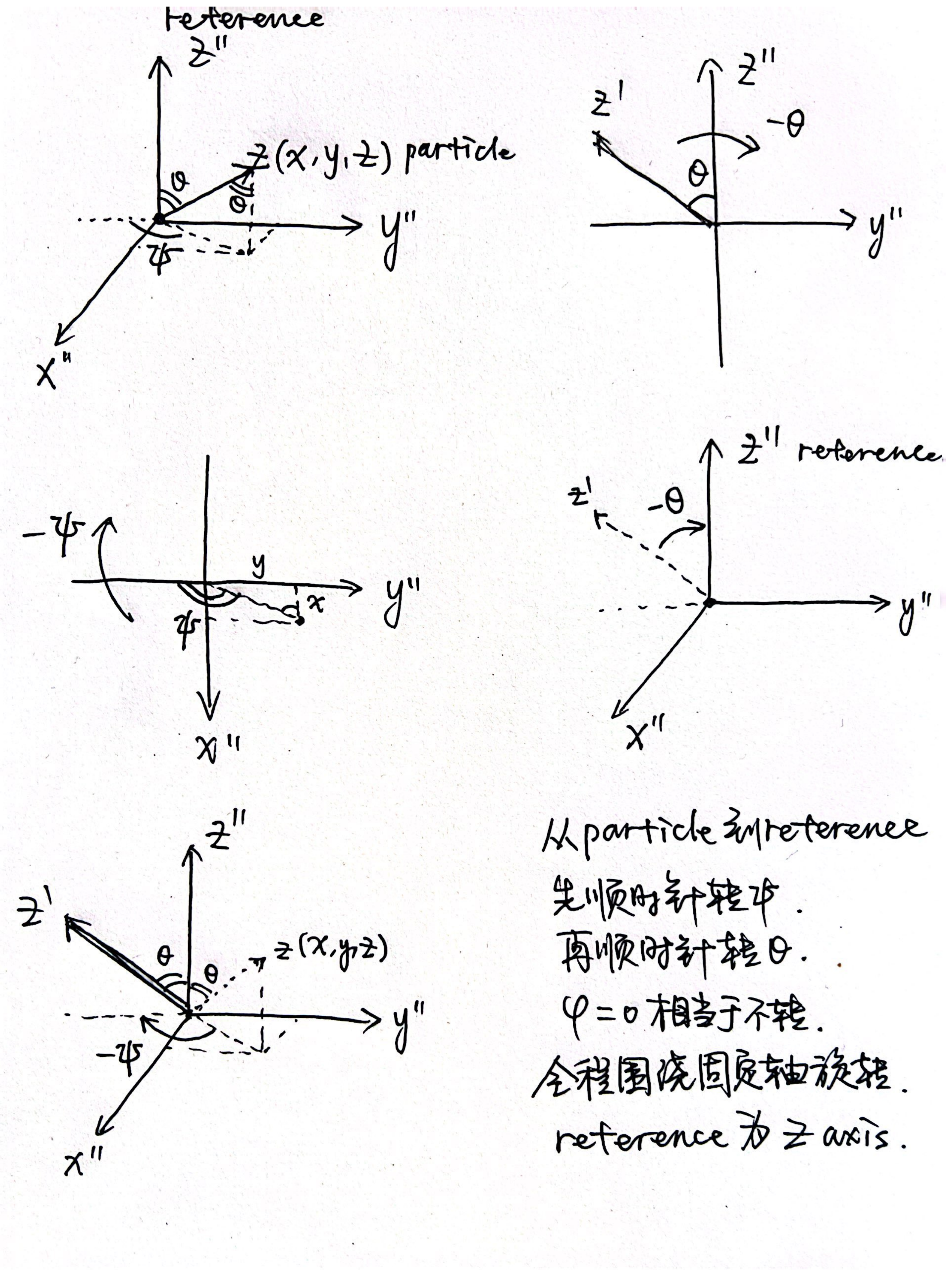
```
# extract and convert eulerangles
eulers_dynamo = table[['tdrot', 'tilt', 'narot']].to_numpy()
eulers_warp = convert_eulers(eulers_dynamo,
                             source_meta='dynamo',
                             target_meta='warp')
data['rlnAngleRot'] = eulers_warp[:, 0]
data['rlnAngleTilt'] = eulers_warp[:, 1]
data['rlnAnglePsi'] = eulers_warp[:, 2]
```

dynamo是外旋zxz

```
euler_angle_metadata = {
    'relion': ConversionMeta(name='relion',
                             axes='zyz',
                             intrinsic=True,内旋，绕新轴
                             right_handed_rotation=True,
                             active=False),

    'dynamo': ConversionMeta(name='dynamo',
                             axes='zxz',
                             intrinsic=False,外旋，绕固定轴
                             right_handed_rotation=True,
                             active=False),
```

dynamo是外旋zxz: 可推断末向量为Z轴

```
disp(['x= ' num2str(cent(1)) 'y= ' num2str(cent(
npart = size(motl,2);
tmp(1:npart) = cent(1);
x = motl(8,:)-tmp;
tmp(1:npart) = cent(2);
y = motl(9,:)-tmp;
tmp(1:npart) = cent(3);
z = motl(10,:)-tmp;
tmp(:) = 90;
theta= 180/pi*atan2(sqrt(x.^2+y.^2),z); %theta
psi  = 90+180/pi*atan2(y,x); %psi
motl(18,:)=psi;
motl(19,:)=theta;
```



先顺时针旋转psi，再顺时针旋转theta，phi取0相当于不转。全程都是围绕固定轴旋转。因此可推断末向量为Z轴。

dynamo是外旋zxz: 可推断末向量为Z轴

生成.em文件时（生成av3的motive list）：第一项phi为0，后两项psi和theta旋转

```
theta= 180/pi*atan2(sqrt(x.^2+y.^2),z); %theta
psi  = 90+180/pi*atan2(y,x); %psi
motl(18,:)=psi;
motl(19,:)=theta;
```

生成.tbl文件时（生成dynamo的table）：前两项tdrot和tilt旋转，最后一项narot为0（真正的外旋zxz）

make_full_table.m | GCBtest_001_object_2.tbl | dynamo2motl.m | motl2dyı

```
0 1 0 0 0 0 -206.57 -159.56 -0 0 0 0 0 0 0 0 0 0 0 1 2 1 0 253 126 86
0 1 0 0 0 0 16.699 -143.29 -0 0 0 0 0 0 0 0 0 0 0 1 2 1 0 280 156 94
0 1 0 0 0 0 45 -168.58 -0 0 0 0 0 0 0 0 0 0 0 1 2 1 0 249 122 95
0 1 0 0 0 0 -74.745 -90 -0 0 0 0 0 0 0 0 0 0 0 1 2 1 0 270 151 81
0 1 0 0 0 0 -81.254 -132.38 -0 0 0 0 0 0 0 0 0 0 0 1 2 1 0 237 124 94
0 1 0 0 0 0 -8.1301 -159.59 -0 0 0 0 0 0 0 0 0 0 0 1 2 1 0 259 133 103
0 1 0 0 0 0 65.225 -132.24 -0 0 0 0 0 0 0 0 0 0 0 1 2 1 0 210 70 98
```

并且tdrot和psi方向相反，tilt和theta方向相反

relion是内旋zyz

Orientations

Orientations (`rlnAngleRot` , `rlnAngleTilt` , `rlnAnglePsi`) in a STAR file rotate the reference into observations (i.e. particle image), while translations (`rlnOriginXAngstrom` and `rlnOriginYAngstrom`) shifts observations into the reference projection. For developers, a good starting point for code reading is `ObservationModel::predictObservation()` in the src/jaz/obs_model.cpp.

In compliance with the Heymann, Chagoyen and Belnap (2005) standard RELION uses a right-handed coordinate system with orthogonal axes X, Y and Z, where right-handed rotations are called positive, and Euler angles are defined as:

- The first rotation is called `rlnAngleRot` and is around the Z-axis.
- The second rotation is called `rlnAngleTilt` and is around the new Y-axis.
- The third rotation is called `rlnAnglePsi` and is around the new Z axis

As such, RELION uses the same Euler angles as XMIPP, SPIDER and FREALIGN.

relion是内旋zyz：初始向量是z轴

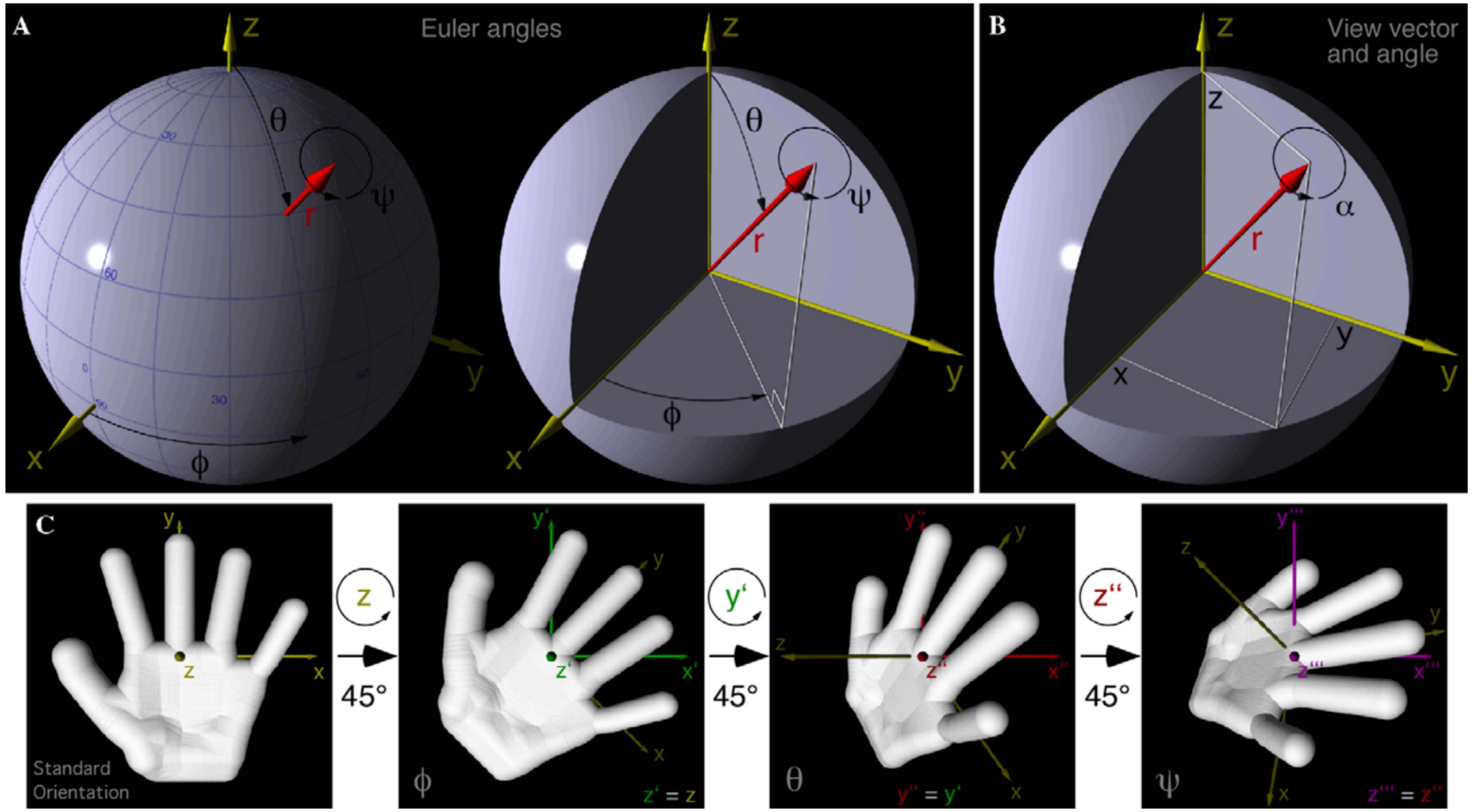


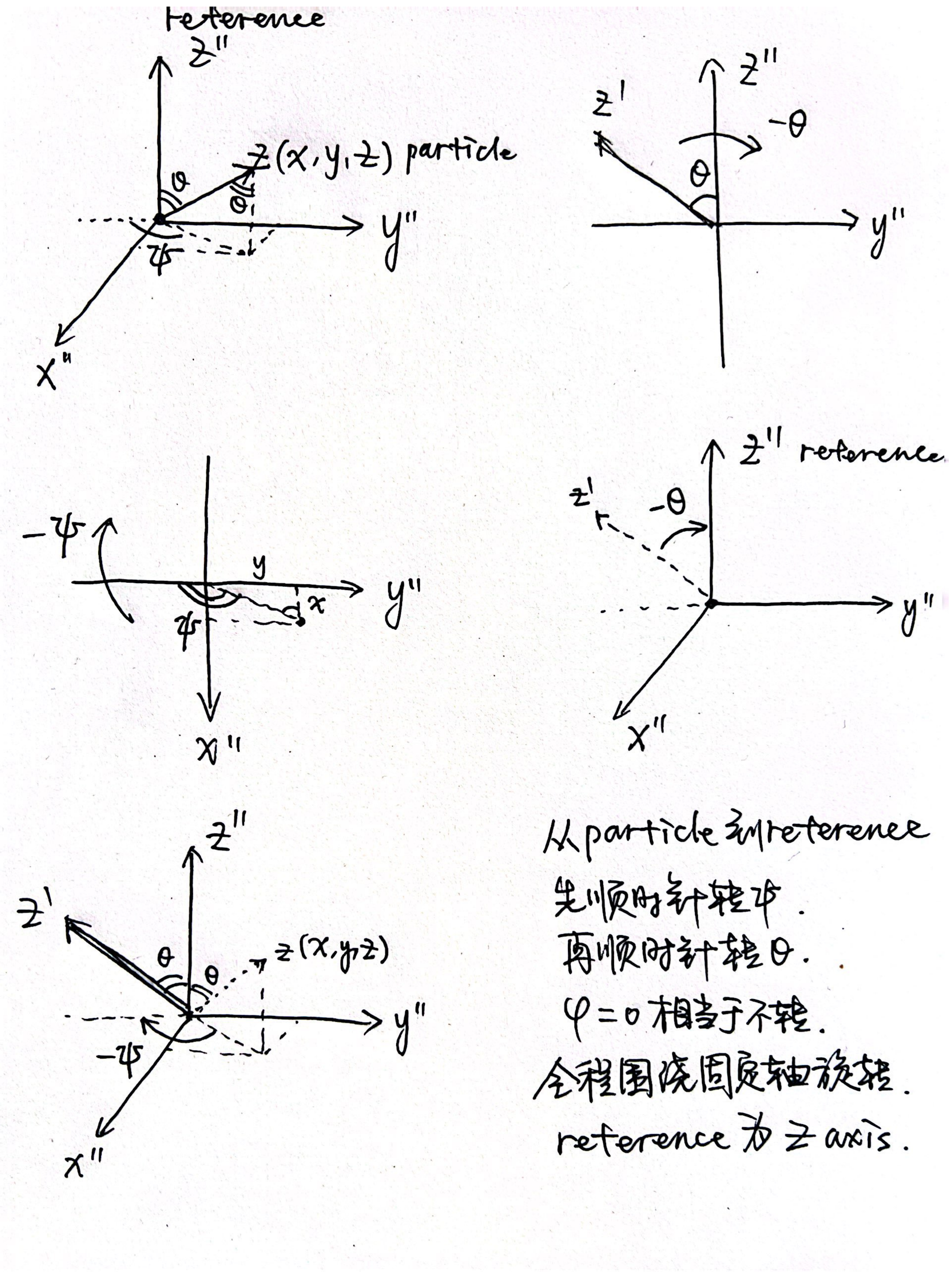
Fig. 1. Definitions of orientation convention about right-handed Cartesian axes **x**, **y**, **z** (yellow) with the origin corresponding to the image origin (origin of the imaged object). (A) *Euler angles*. The position of any vector beginning at the origin can be described by the angles ϕ and θ (in this example, $\phi = \theta = 45°$) The angle ϕ is the angle measured in the anti-clockwise direction between a projection of the vector **r** onto the **x**–**y** plane and the positive **x**-axis. (ϕ is analogous to longitude.) The angle θ is the angle between the vector and the positive **z**-axis. (θ is analogous to latitude.) Rotation about the vector is described by the angle ψ (analogous to compass heading). (B) *View vector and angle*. The vector **r** is the same as the corresponding vector in panel (A) **r** is described by coordinates $\{x, y, z\}$ (labeled in black) in each of the three Cartesian axes **x**, **y**, **z**. The view angle (α) is a rotation about **r**. (C) *Rotation matrices for Euler angles*. A model of a human left hand is used to demonstrate the three angles. As applied in rotation matrices, the convention involves three right-handed rotations about successive orthogonal axes: first, rotation about **z** by ϕ; second, rotation about **y**′ by θ; and third, rotation about **z**″ by ψ. The standard axes {**x**,**y**,**z**} are shown in yellow in each frame. The prime, double-prime, and triple-prime axes are shown in green, red, and magenta, respectively. Note, when the axis of rotation is pointing at the viewer, the coordinate system is rotated anti-clockwise and the object clockwise. The program POV-Ray was used in making this figure (http://www.povray.org).

particle to reference还是reference to particle

```
# Created by the starfile Python package (version 0.4.11) at 19:23:18 on 02/04/2023

data_

loop_
_rlnTomoName #1
_rlnTomoParticleId #2
_rlnTomoManifoldIndex #3
_rlnCoordinateX #4
_rlnCoordinateY #5
_rlnCoordinateZ #6
_rlnOriginXAngst #7
_rlnOriginYAngst #8
_rlnOriginZAngst #9
_rlnAngleRot #10
_rlnAngleTilt #11
_rlnAnglePsi #12
_rlnClassNumber #13
_rlnRandomSubset #14
GCB_004 1	1	1620.000000	774.000000	378.000000	0	0	0	0.000000	146.050000	-158.199000	1	1
GCB_004 2	1	1578.000000	756.000000	396.000000	0	0	0	0.000000	143.760000	82.875000	1	1
GCB_004 3	1	978.000000	504.000000	444.000000	0	0	0	0.000000	161.570000	180.000000	1	2
GCB_004 4	1	1146.000000	576.000000	444.000000	0	0	0	0.000000	139.860000	-161.565000	1	2
GCB_004 5	1	1074.000000	540.000000	468.000000	0	0	0	0.000000	146.350000	108.435000	1	1
GCB_004 6	1	1140.000000	480.000000	468.000000	0	0	0	0.000000	155.790000	-20.560000	1	1
GCB_004 7	1	1116.000000	582.000000	480.000000	0	0	0	0.000000	157.860000	-159.444000	1	2
GCB_004 8	1	1092.000000	516.000000	510.000000	0	0	0	0.000000	164.500000	33.690000	1	1
GCB_004 9	1	2124.000000	930.000000	534.000000	0	0	0	0.000000	178.090000	0.000000	1	1
GCB_004 10	1	936.000000	516.000000	378.000000	0	0	0	0.000000	105.020000	116.565000	1	1
GCB_004 11	1	1902.000000	888.000000	474.000000	0	0	0	0.000000	167.760000	-104.036000	1	2
```

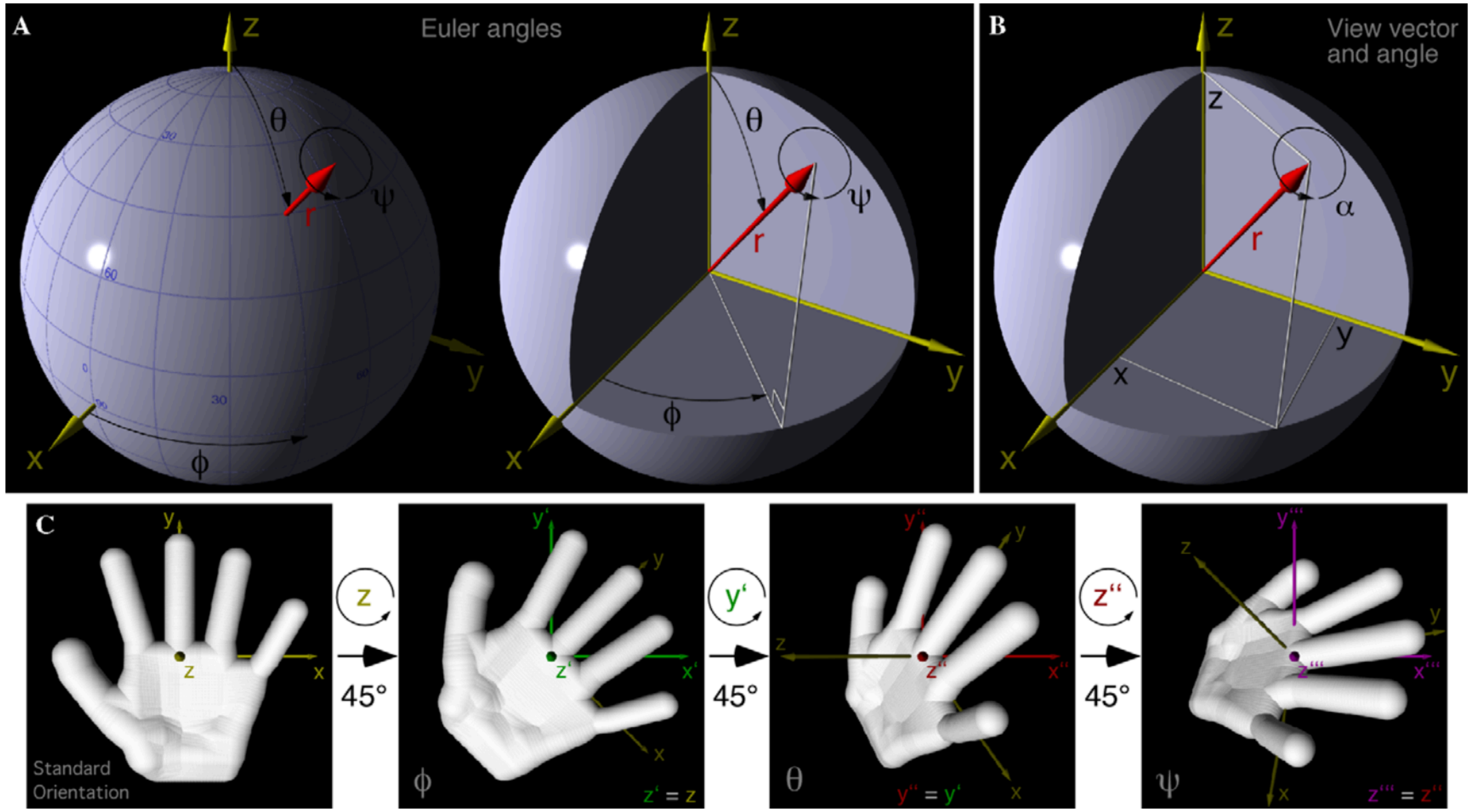
RELION是reference转到particle

particle to reference还是reference to particle



DYNAMO是particle转到reference

relion是reference to particle



需要特别注意的是，对于relion而言，目前整个tomogram的坐标系是x'''y'''z'''，而particle的朝向是z，因此在计算euler angle时，实际particle的坐标是在x'''y'''z'''坐标系下的，而particle朝向是z。