GPU ISAC (Chimera version) installation notes

NOTE: GPU ISAC requires the installation of the SPHIRE package

Installation steps:

(a) Open a terminal and enter the folder where you put the `GPU_ISAC_CHIMERA.tar` file.

(b) Untar the .tar archive:

```
tar -xf GPU_ISAC_CHIMERA.tar
```

(c) Check CUDA path variables:

```
echo $PATH
```
does it contain the path to your **cuda/bin** folder?

```
echo $LD_LIBRARY_PATH
```
does it contain the path to your **cuda/lib64** folder?

If these path variables do **not** contain the cuda/bin and cuda/lib64 folder, add then like so:

```
export PATH=/path/to/cuda/bin:${PATH}
export LD_LIBRARY_PATH=/path/to/cuda/lib64:${LD_LIBRARY_PATH}
```

Where `path/to/cuda/bin` and `path/to/cuda/lib64` need to be replaced with the real paths to the respective folders. If you do not know where to find them, by default they should be located in `/usr/local/cuda`.

(d) Compile the C++/CUDA library needed for GPU ISAC.:

```
cd vChimera/cuda
nvcc gpu_aln_common.cu gpu_aln_noref.cu -o gpu_aln_pack.so -shared -Xcompiler -fPIC -lcufft -std=c++11
```

(e) Adjust sparx libraries to work with the CUDA library we just compiled:

```
cd ../eman2/sparx/libpy
```
(we are now in the **vChimera/eman2/sparx/libpy** folder)

```
sed -i.bkp "s|/home/schoenf/work/code/cuISAC/cuda|$(realpath ../../../cuda)|g" applications.py

sed -i.bkp2 's|statistics.sum_oe( data, "a", CTF, EMData(), myid=myid|statistics.sum_oe( data, "a", CTF, EMData()|g' applications.py
```

(f) Tell GPU ISAC to use the correct libraries and environment:

```
cd ../bin
```
(we are now in the **vChimera/eman2/sparx/bin** folder)

```
ln -rs ../libpy/* .

sed -i.bkp "s|/home/schoenf/applications/sphire/v1.1/envs/sphire_1.3/bin|$(dirname $(which sphire))|g" sxisac2_gpu.py

sed -i.bkp2 "s/^\(.*options, args.*\)$/\1\n    os.environ['CUDA_VISIBLE_DEVICES'] = options.gpu_devices\n    options.gpu_devices = ','.join(map(str, range(len(options.gpu_devices.split(','))))/g" sxisac2_gpu.py
```

(see next page for an example)

Example use (replace path/to/isac):

```
mpirun -np 6 /path/to/sxisac2_gpu.py bdb:path/to/stack out_dir --CTF –radius=160 --target_radius=29 --target_nx=76 --img_per_grp=100 --minimum_grp_size=60 --thld_err=0.7 --center_method=0 --gpu_devices=0,1
```

MANDATORY parameters:

- Replace "`/path/to/sxisac2_gpu.py`" with the path to your **sxisac2_gpu.py** file.
- Replace "`path/to/stack`" with the path to your **input .bdb stack**. If you are using an **.hdf** stack, you need to remove the "`bdb:`" part of the command.
- Replace "`out_dir`" with the path to your preferred **output directory**.
- Adjust the number in "`--radius=160`" to the **radius of your particle** (in pixel).

OPTIONAL parameters:

- In "`mpirun –np 6`" the number can be set to the number of your **CPU processors** (e.g., if you have a quad core CPU, you would use 4 here).
- In "`--gpu_devices=0,1`" you can set **what GPUs to use**. This example uses two GPUs with id values 0 and 1, respectively. You can check the id values of your available GPUs by executing "`nvidia-smi`" in your terminal (GPUs are listed by capability, with 0 being your strongest GPU).