Collection of ideas for short-term (max 6 months) prototypes to be worked on in 2020.

Urgent, needed for other R&Ds
Normal priority
Second priority, possible GSOC or summer student projects

Improve, optimise and modernise existing Geant4

1) Complete the modification in Geant4 to achieve 'stateless' transport prototype. The final goal is to be able to implement a Geant4 prototype demonstrating some form of code locality (multiple tracks to a single kernel). This is also a step towards more code modularity, cutting code dependencies and enabling a functional programming approach. Estimated effort: 1-2 months FTE (Witek, Andrei) for the first working demonstrator

2) Automate performance measurements to be run in continuous integration to prevent performance regressions, as well as assess improvements made by a pull request
    a) Hotspots and microarchitecture exploration data
    b) Important metrics (e.g. instruction/data cache misses per function, GFLOPs)
    c) Integration in geant-val or with grafana if possible

    Estimated effort: 2-3 month FTE (Guilherme)

3) Single precision usage in simulation components. Study numerical stability issues and changes needed to overcome divergence for geometry, physics models, magnetic field propagation. Using single instead of double precision at least in the state representation would allow to better fit data caches. For vectorized algorithms this implies also doubling the number of possible simultaneous operations. Estimated effort: 2-3 months FTE per category (Andrei, summer student, …)

4) Instruction and data cache optimizations
    a) Reorder data members in Geant4 hot classes by size to reduce padding
    b) Group booleans stored as G4bool into bit fields
    c) Optimize maximum size of in-lined functions, compare -O2 with -Os
    d) Study potential improvements with link-time and profile guided optimizations

    Estimated effort: 1-2 weeks FTE for changes + performance measurements (Guilherme)

5) Investigation of alternative approach to TLS currently used in Geant4. Evaluate the performance impact of the current use of TLS. Prototyping of TLS-free version of Geant4. Estimated effort: 2-3 month FTE

6) Multi-track pipelining of the stepping loop: advancing memory-coherent track states by passing them through the sequence of computing kernels, one kernel at a time using masking. This sequencing is much more GPU-friendly, while on CPU aims to demonstrate simultaneous code and data locality. The idea is fastest to test in the GeantV prototype that is stateless. Estimated effort: 1-2 month FTE for a first demonstrator (Andrei)

7) Implement GJK algorithm in VecGeom as alternative for all convex shapes. Estimated effort: 1-2 month FTE

8) Transforming key geometry regions to simpler description. Many important detector volumes are composed of a mother volume in which a large number of daughter volumes is placed without filling it (e.g. ATLAS Ecal volumes.) This means that the geometry modeller must involve many daughter volumes when locating a point, or when computing the next intersection of a track (or ray). This project will investigate the potential for breaking up the 'remainder' of mother volumes into small 'subvolume' pieces for which navigation would be straightforward -- and for which the volume on the other side of their surfaces could be identified at construction or precalculated.
    a) Finalizing analytic neighbor detection algorithm (most of the implementation existing already)
    b) Implementing navigator specialization for such cases

   Estimated effort: 4 months (Andrei, …)

9) Implement volume-preserving symplectic integrator as replacement for Runge-Kutta. See, e.g., https://aip.scitation.org/doi/10.1063/1.4962677 for more information. Estimated effort: 1-3 month FTE (a good project for GSOC of Summer Student)

Fast simulation techniques

1) Classical parameterization of EM showers. Extending existing functionalities of G4 fast simulation (GFlash model). Automated procedure of tuning parameters to the specific geometry. Identify and include extra functionalities included in i.a. the CMS parameterization. Estimated effort: 3 - 4 months (Anna)

2) ML-aided shower parameterization. Investigation of autoregressive networks on a simple geometry. Draw conclusions on the potential of the autoregressive models for the simulation of calorimeters. Estimated effort: 4 - 5 months (Ioana)

3) Inference within G4. Integrate inference library into G4 toolkit to allow the usage of a trained network within the simulation. Formulate conclusion on generality of the approach and if/how other NNs can be used with those tools. Estimated effort: 1 month (Ioana, Anna)

4) Validation tools for fast simulation of showers. Common tools for validation of the fast simulation: Shower profiles, first/second moments, energy response, cumulated and/or per detector layer, simulation time, … To be shared between any fast simulation approach (classical/ML-aided). Estimated effort: 2 months (Anna, Ioana)

Use of accelerators

1) GPU-friendly geometry modeler and navigator
   Prototype a geometry modeler to navigate and calculate the safety on the GPU. Approaches to be tested:
    - Compare and study state of existing solution (VecGeom has basic navigation on GPUs)
    - Usage of limited number of primitives (e.g. all faceted solids represented as tessellations) to open up single particle - multiple shapes/triangles parallelism
    - Usage of bounding box acceleration structures, ported from VecGeom sequential implementation.
    - Benchmark different situations:
        - Many particles in same volume
        - Many particles but each having a different touchable state (navigation state)
    - Investigate use of reduced number of primitives (e.g. all 6-faceted solids represented as general type instead of box, para, …) to reduce GPU code divergence.

   Estimated effort: 6 months FTE (Andrei, …)

2) Evaluation of creating geometries for vendor optimised GPU package. Libraries for GPU ray tracing simulation are optimised for GPU hardware, and can be applied most easily for simulation of optical photons. The package Optix offloads the simulation of optical photons from Geant4 using specialised simulation based on Nvidia Optix. This task will evaluate the potential of this and similar solutions for specific geometry/-ies, starting with the LHCb RICH geometry which is a leading consumer of CPU in LHCb simulation. It will

in particular investigate the geometry construction options and their strengths and limitations. Estimated effort: 3-5 months (FTE in 1-2 persons) (Andrei, …)

3) Evaluate 'performance portability' framework using sim kernel(s). Evaluate the leading 'performance portability' frameworks (Kokkos, Alpaka or SYCL/oneApi) for use in detector simulation by porting a small code base with medium complexity to use it. The kernel preferably should include a challenge in the form of code that involves divergent paths of execution (or this could be stage 2.)
   Potential candidate 'kernels' (only one must be chosen as target)
   - Step computation in voxel geometry
   - integration of tracks in pure magnetic field
   - Restricted set of physics for one particle type in one medium, which choice of 2 processes (models.)
   Estimated effort: 3-4 months per framework

4) Geometry transformations exposing massive parallelism
   For example: single solid type (polyhedron, tetrahedra), or tessellations (everything made of triangles)