Beam Transport & Acceleration with Space Charge with the Halodyn Code

Andrea Franchi
ESRF, Grenoble

HPC@CNAF meeting, Bologna, 24 June 2014

Outline

- a bit of hystory
- the Halodyn code
- performances, comparisons and applications

Halodyn: a bit of history

- end of 90s: 2D & 3D transport through a FODO multi-cell with space charge (2D & 3D PIC Poisson solver) [Turchetti & Rambaldi]
- 1999-2000: transport parallelized via Message-Passing Interface (MPI) tested and run at CINECA [Turchetti]
- 2001-2002: 3D version called Halodyn for an arbitrary linac lattice with acceleration [Franchi]
- 2002: MPI version of Halodyn tested in Bologna & INFN-Legnaro; massive simulations with 10^6 macro-particles of the Legnaro's TRASCO DC 30mA 30MW proton linac
- 2003-2006: Systematic benchmarking and comparison of Halodyn in the frame of the CARE network HIPPI (GSI)
- 2006: improving GSI UNILAC DTL transport with Halodyn

The Halodyn code

- Flexible input file for an arbitrary linac structure compatible with Los Alamos Nat. Labs' PARMILA code
- FORTRAN77 2D & 3D Particle-In-Cell Poisson solver for the computation of the space-charge electric field (closed boundary conditions, arbitrary contourn)
- FORTRAN77 Beam transport (2nd-order symplectic integrator) parallelized via MPI, good CPU time scaling $\propto 1/N_p$ (Poisson solver not parallelized)
- 3D focusing matching with space charge (envelope equation)
- acceleration through thin RF cylindrically symmetric cavities
- several types of initial multi-particle distributions & from external file (for comparisons)
- C++ Graphical post-processing tools

The Halodyn code: the Poisson Solver

- **Particle-In-Cell solver with 2 charge deposition options**
- **3D FFT algorithm [complex. $\propto (K \log_2 K)^3$] or 2D FFT + 3-diagonal inversion [complex. $\propto K^3(\log_2 K)^2$], K number of cell nodes per dimension**
- **Closed (Dirichelet) boundary conditions over the 3D box (test version over cylinder with arbitrary cross-section)**
- **Space charge electric field via linear interpolation**

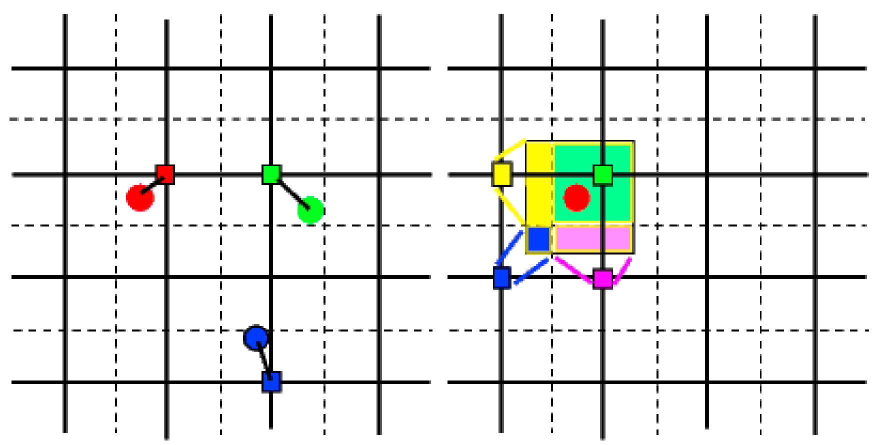


Figure 1.1: 2D NGP (left) and 2D CIC (right) method.

The Halodyn code: the Poisson Solver

- Particle-In-Cell solver with 2 charge deposition options
- **3D FFT algorithm [complex. $\propto (K \log_2 K)^3$] or 2D FFT + 3-diagonal inversion [complex. $\propto K^3(\log_2 K)^2$], K number of cell nodes per dimension**
- Closed (Dirichelet) boundary conditions over the 3D box (test version over cylinder with arbitrary cross-section)
- Space charge electric field via linear interpolation

$$\begin{pmatrix} \rho \\ V \end{pmatrix} = \sum_{\boldsymbol{k}} \begin{pmatrix} \rho_{\boldsymbol{k}} \\ V_{\boldsymbol{k}} \end{pmatrix} \sin\left(\frac{\pi}{L_x} k_x x\right) \sin\left(\frac{\pi}{L_y} k_y y\right) \sin\left(\frac{\pi}{L_z} k_z z\right)$$

$$\Delta V(x, y, z, s) = -4\pi\rho(x, y, z, s), \quad \rightarrow \quad V_{\boldsymbol{k}} = \frac{4}{\pi}\rho_{\boldsymbol{k}}\left(\frac{k_x^2}{L_x^2} + \frac{k_y^2}{L_y^2} + \frac{k_z^2}{L_z^2}\right)^{-1}$$

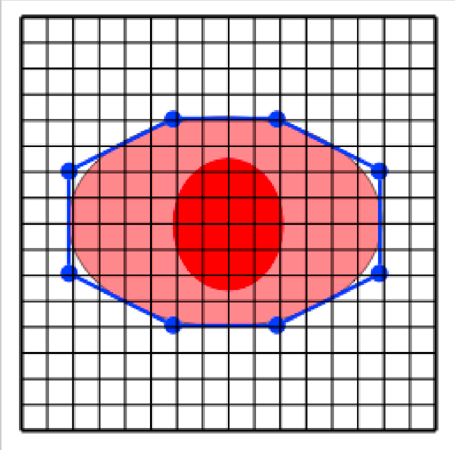
$$E_x(q_x, q_y, q_z) = -\frac{V(q_x + 1, q_y, q_z) - V(q_x, q_y, q_z)}{2\Delta x}$$

The Halodyn code: the Poisson Solver

- Particle-In-Cell solver with 2 charge deposition options
- 3D FFT algorithm [complex. $\propto (K \log_2 K)^3$] or 2D FFT + 3-diagonal inversion [complex. $\propto K^3(\log_2 K)^2$], K number of cell nodes per dimension
- **Closed (Dirichelet) boundary conditions over the 3D box (test version over cylinder with arbitrary cross-section)**
- Space charge electric field via linear interpolation

$$V_0(\vec{x}_j) + \sum_k \alpha_k G_k(\vec{x}_j) = 0$$

$$\rho'(\vec{x}) = \rho(\vec{x}) - \sum_k \alpha_k \delta_{\vec{x},\vec{x}_k}$$



The Halodyn code: the Poisson Solver

- **Particle-In-Cell solver with 2 charge deposition options**
- **3D FFT algorithm [complex. $\propto (K \log_2 K)^3$] or 2D FFT + 3-diagonal inversion [complex. $\propto K^3(\log_2 K)^2$], K number of cell nodes per dimension**
- **Closed (Dirichelet) boundary conditions over the 3D box (test version over cylinder with arbitrary cross-section)**
- **Space charge electric field via linear interpolation**

The Halodyn code: the parallel version

- **Only the beam transport is parallelized via MPI. Good CPU time scaling unless high grid resolution is used**

particelle	1 CPU	5 CPU	10 CPU	20 CPU
10^4	3	1.6 (6.3)	1.6 (10.9)	1.8 (20.8)
10^5	25.2	6.2 (28.7)	3.9 (33.6)	2.9 (43.9)
2×10^5	50.6	11.2 (53.4)	6.4 (58.4)	4.0 (68.9)
3×10^5	75.5	16.7 (78.7)	8.9 (83.1)	5.3 (93.3)
7×10^5	179.3	36.9 (181.1)	19.4 (186.6)	13.6 (198.4)
10^6	256.2	52.8 (257.8)	27.3 (519.6)	14.5 (274.6)
2×10^6	511.6	103.7 (512.4)	52.7 (519.6)	27.4 (528.6)

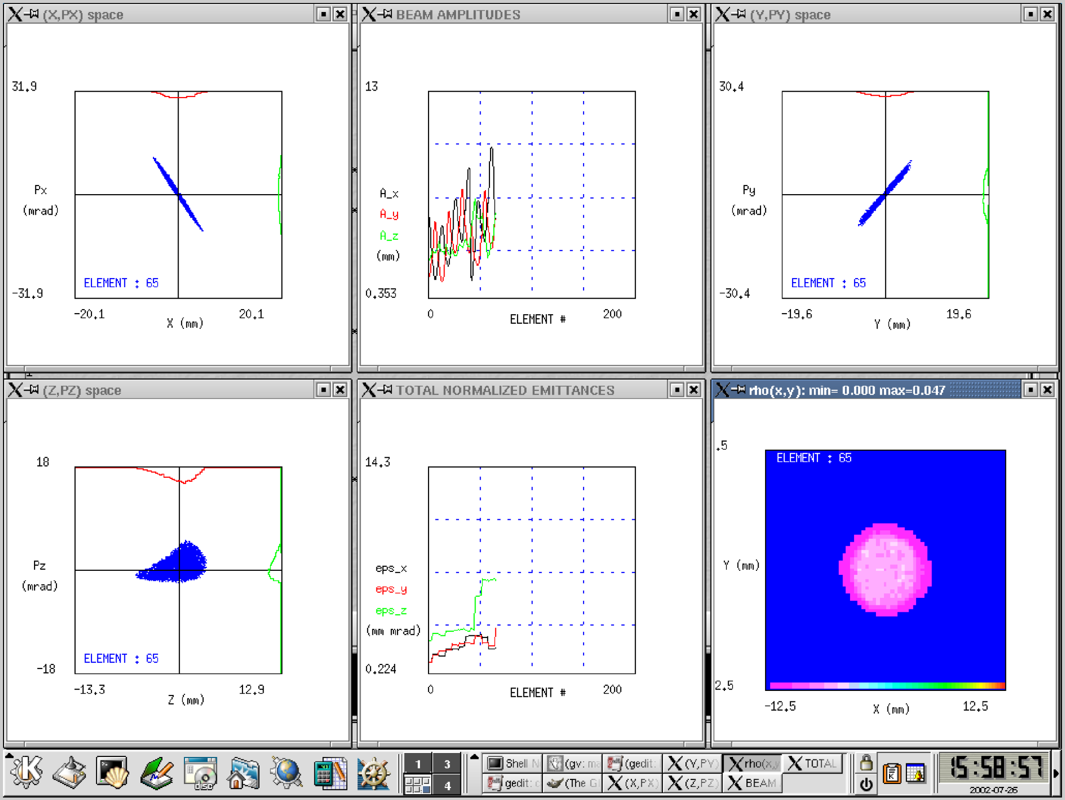
Tabella 3.4: Tempi solari (in s) per un periodo al variare delle CPU e del numero di particelle. Risoluzione a 64x64. Tra parentesi i tempi di CPU

particelle	1 CPU	5 CPU	10 CPU	20 CPU
10^4	11.7	13.6 (65.3)	14.2 (136.9)	15.0 (286.3)
10^5	36.2	18.5 (89.6)	16.7 (161.1)	16.2 (311.3)
2×10^5	64.2	25.0 (128.1)	19.9 (191.1)	19.2 (344.4)
3×10^5	91.4	30.1 (145.5)	23.0 (218.4)	19.4 (367.8)
7×10^5	201.0	52.3 (255.8)	33.8 (327)	24.8 (454)
10^6	282.6	68.8 (337.7)	42.3 (411.7)	29.6 (565.7)

Tabella 3.6: Come sopra, con risoluzione 256x256

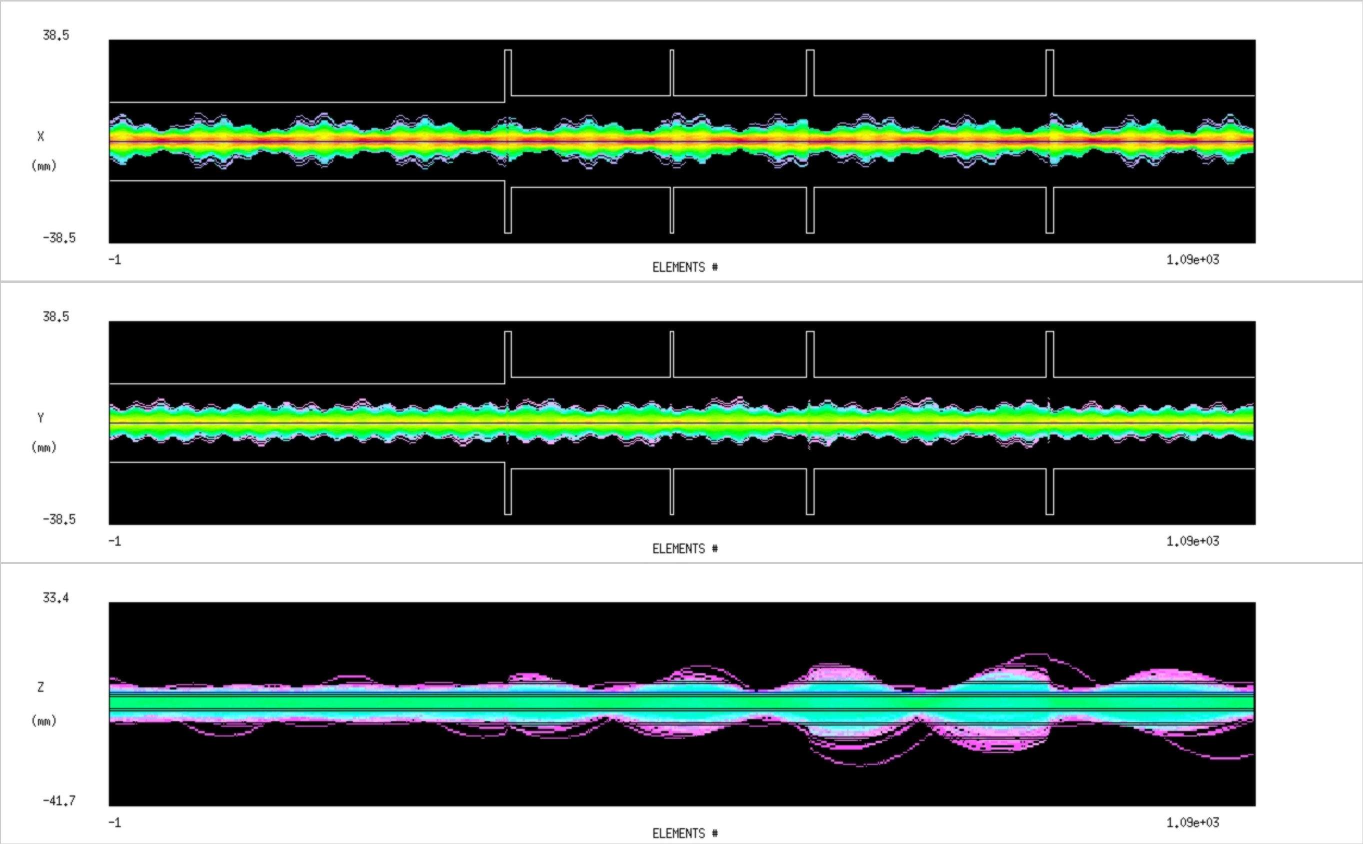
The Halodyn code: the post-processing

- GUI application to display evolution of beam distribution and parameters along the structure



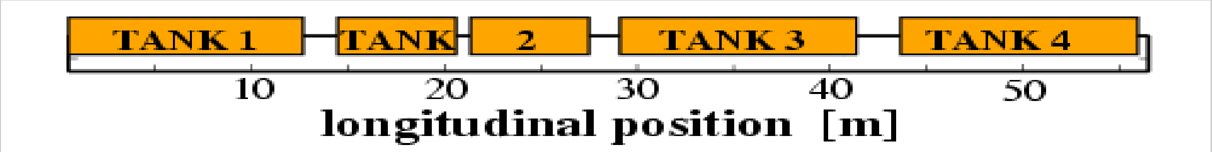
The Halodyn code: the post-processing

- GUI application to display evolution of beam distribution and parameters along the structure



The Halodyn code: performances

GSI UNILAC DTL section: tracking with space charge



$^{238}U^{28+}$, $I_b = 37.5$ mA, T=1.4→11.4(30) MeV/u

# of macrop.	code name [instit.]	CPU time	S-C solver
10^6	IMPACT* [Los Alamos / Berkley]	~4 days	(3D)
	HALODYN* [Bologna]	~1 day	(3D)
	PARTRAN [Saclay]	~6 days	(3D)
10^5	PARMILA [Los Alamos]	~1.5 h	(2D)
	PATH [CERN]	~1.5 h	(2D)
2×10^4	PATH [CERN]	~1.5 days	(P-P)
5×10^3	DYNAMION [GSI]	~1.3 days	(P-P)

$*$: to be scaled with # of CPU's

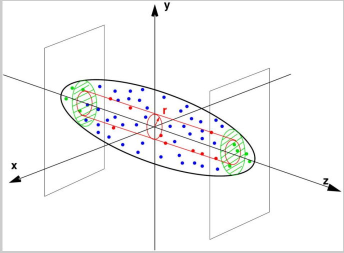
3D : x-y-z PIC solver

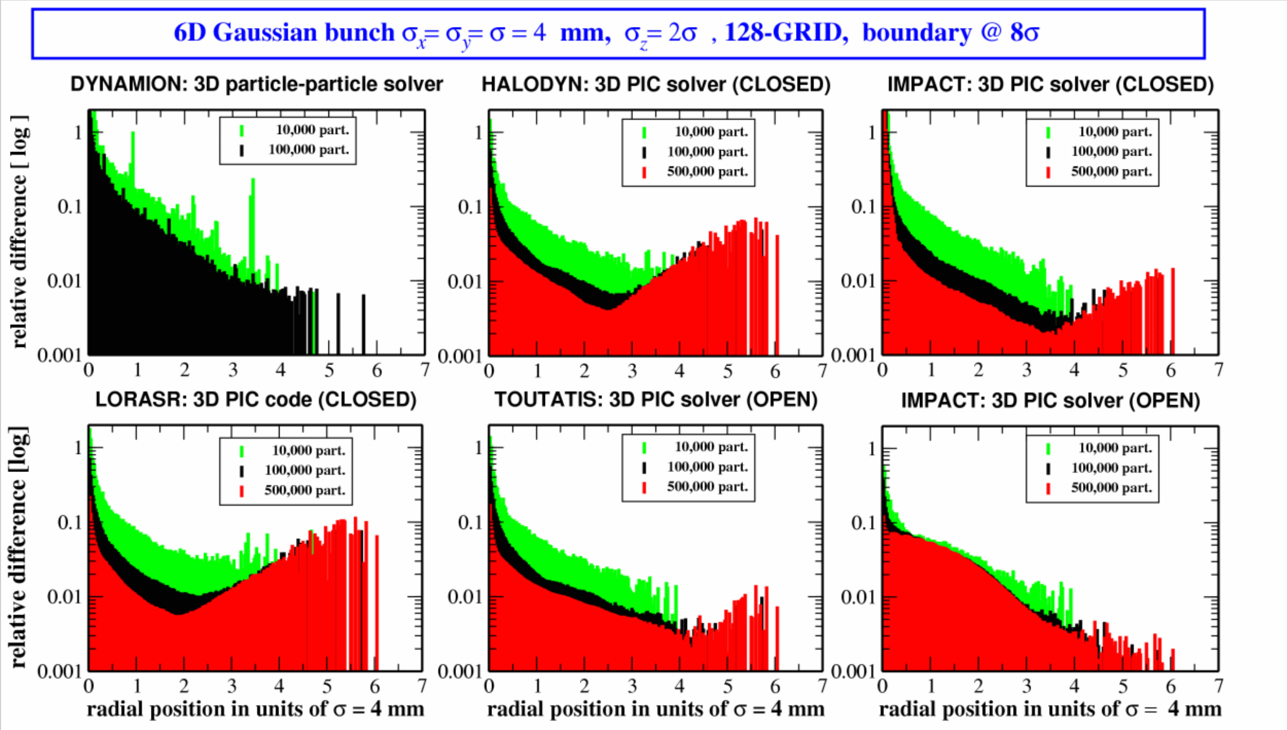
2D : r-z azimuthally symmetric PIC solver

P-P: direct particle-particle Coulomb interaction

Code benchmarking (WP-5 CARE-HIPPI network)

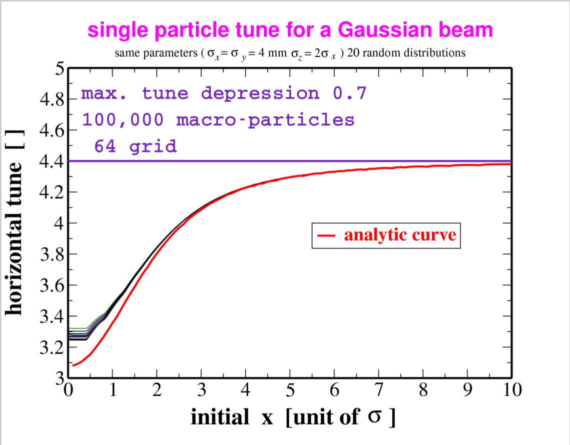
Solver accuracy: $\vec{E}_{SC}$ field from given distribution (implicit open boundary conditions)

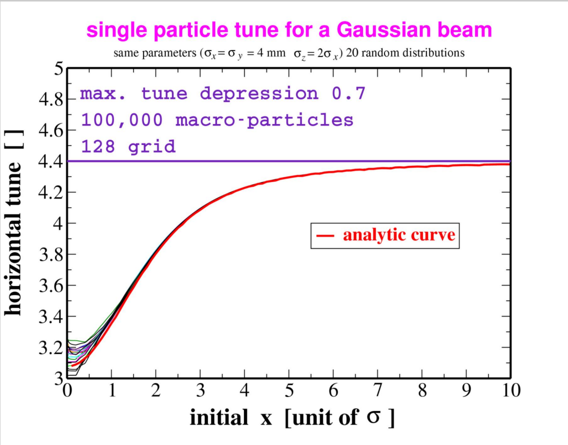


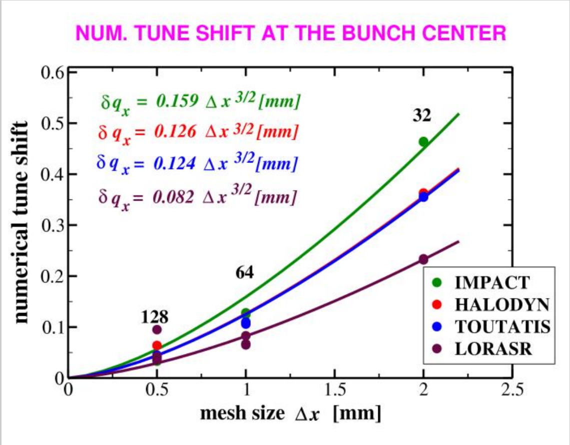


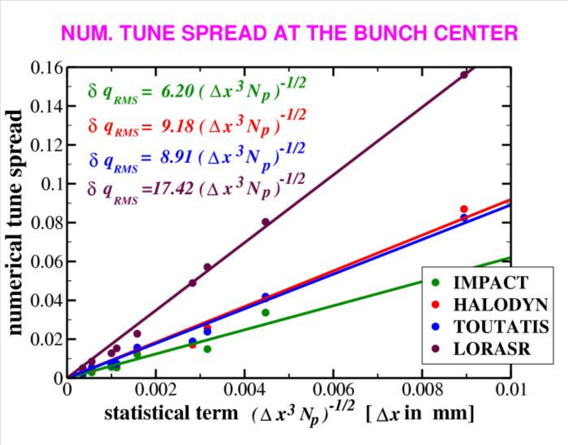
Code benchmarking (WP-5 CARE-HIPPI network)

Solver accuracy: tune shift from given distribution



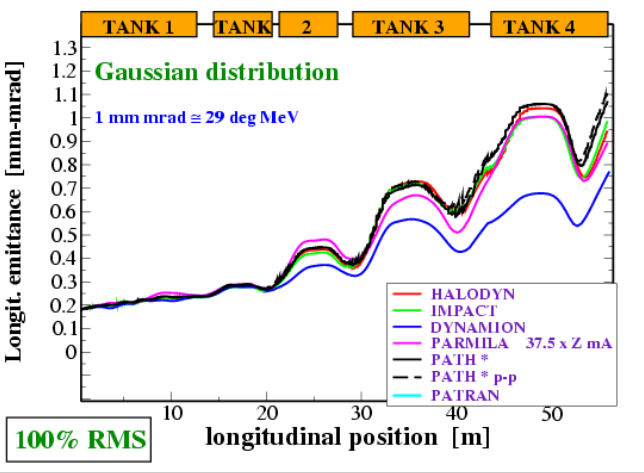


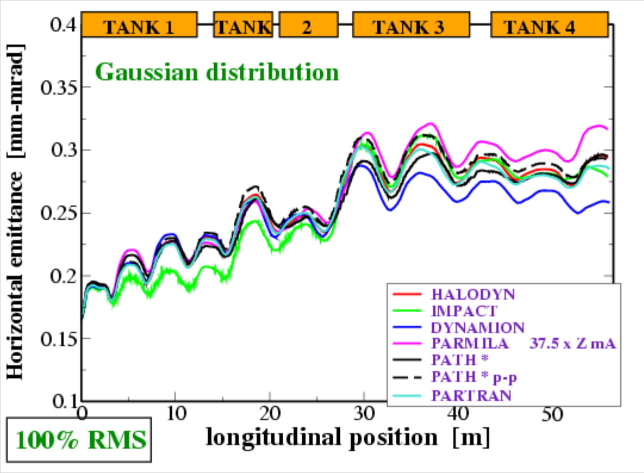




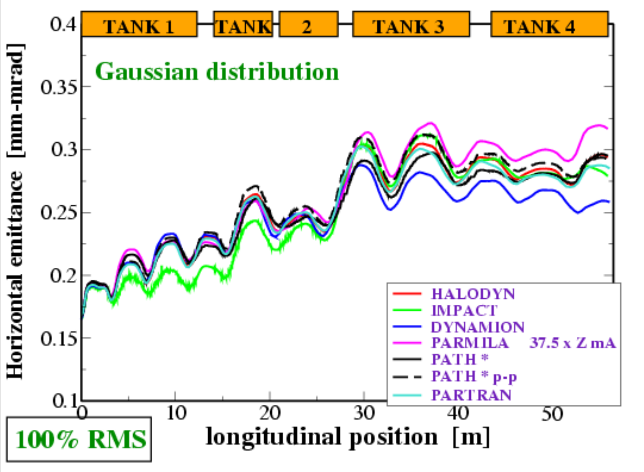
Code benchmarking (WP-5 CARE-HIPPI network)

GSI UNILAC DTL tracking with space charge

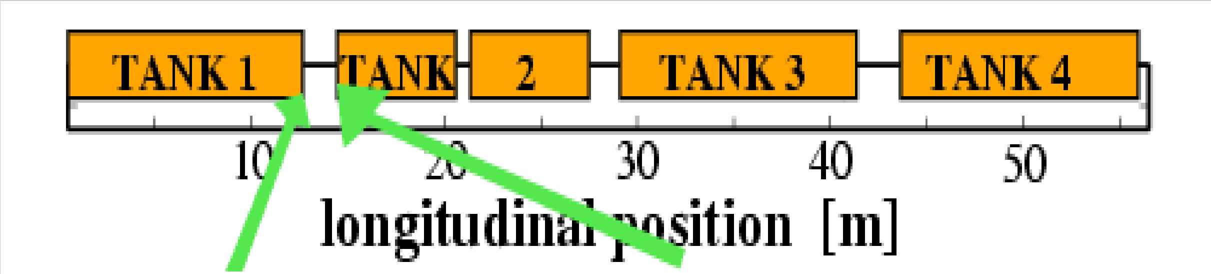




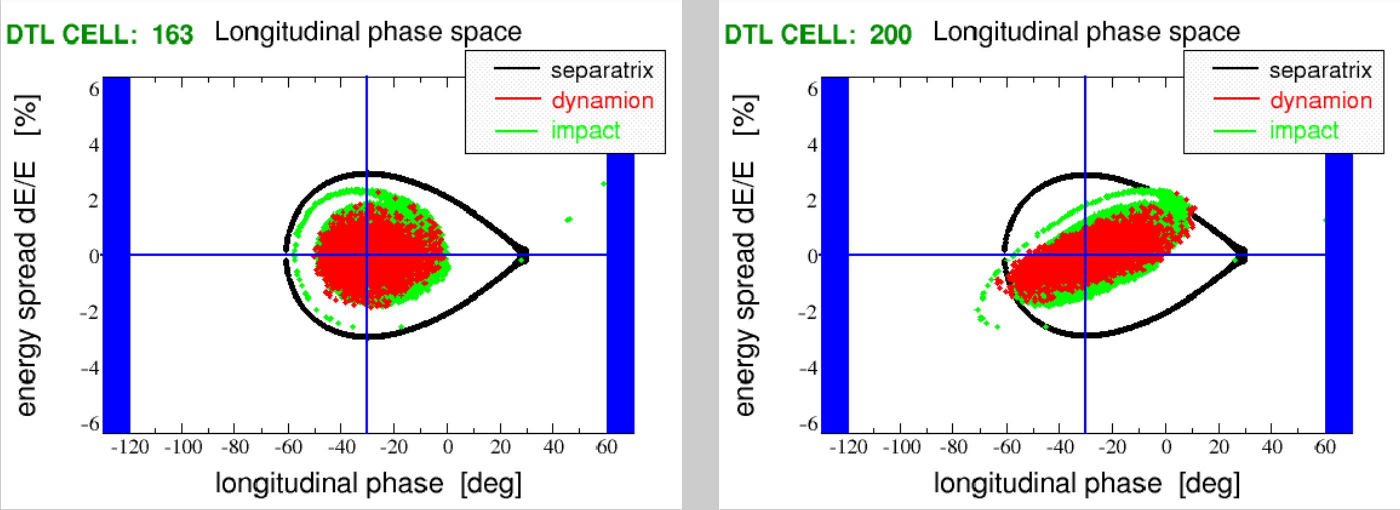
click me (small init. ϵ_z)
click me (large init. ϵ_z)



Halodyn as matching tool for the GSI UNILAC DTL



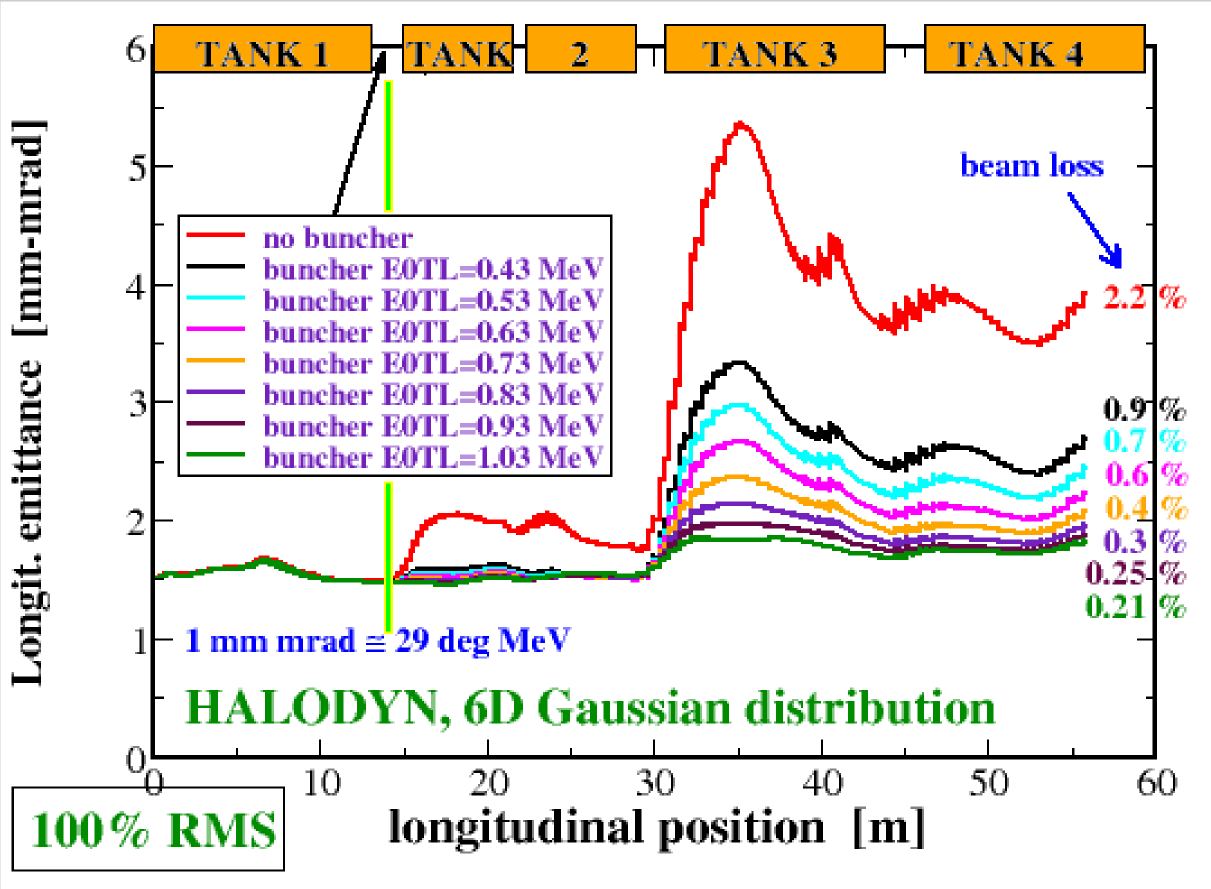
exit tank A1 | entrance tank A2



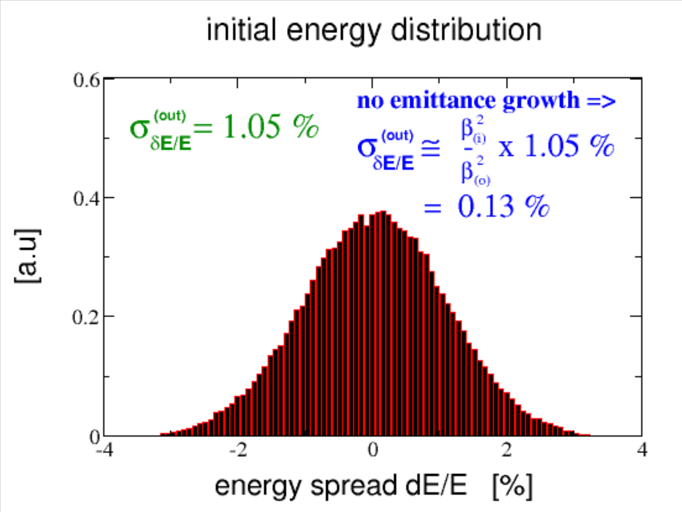
No buncher cavity between tanks A1-A2 (~ 1.5 m $\sim 8\beta\lambda$) $\Rightarrow$
beam enters tank A2 longitud. mismatched + SC $\Rightarrow \epsilon_z$ **growth**

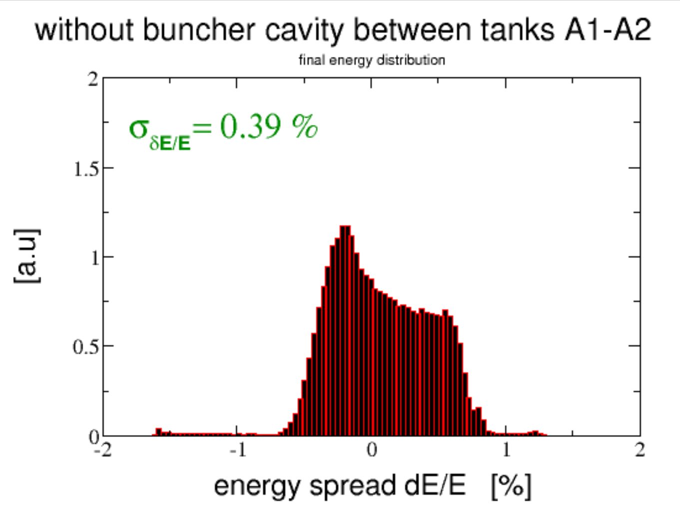
Halodyn as matching tool for the GSI UNILAC DTL

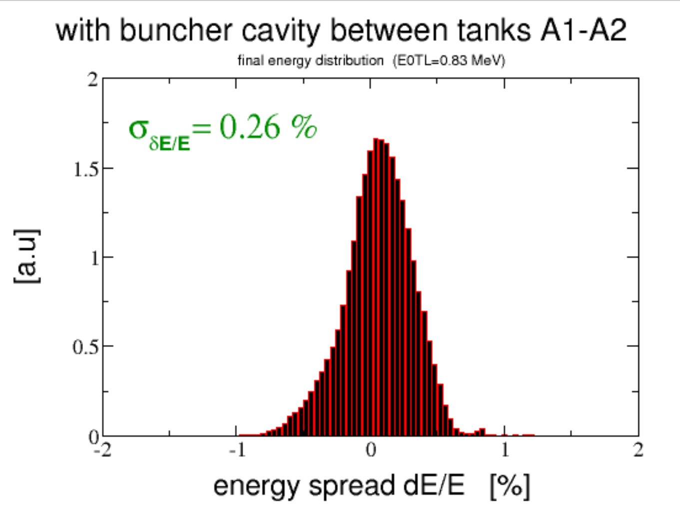
Longitudinal emittance Vs. gradient of a proposed buncher cavity



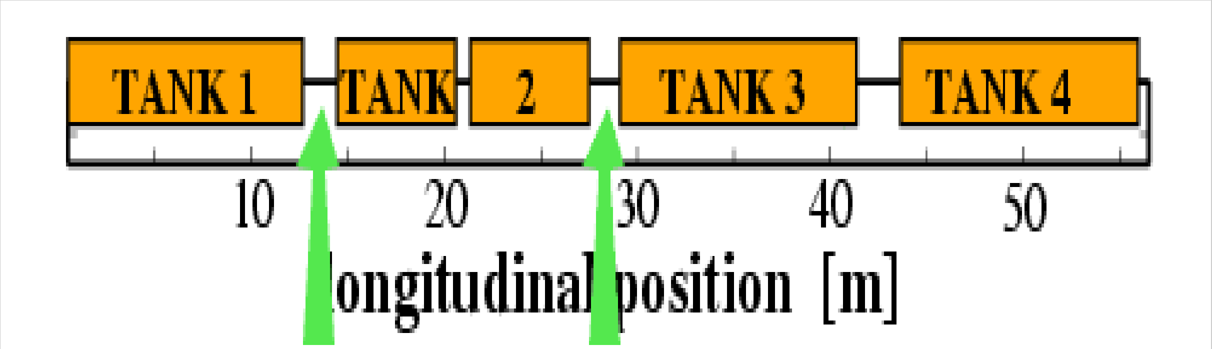
Energy distribution with and without buncher





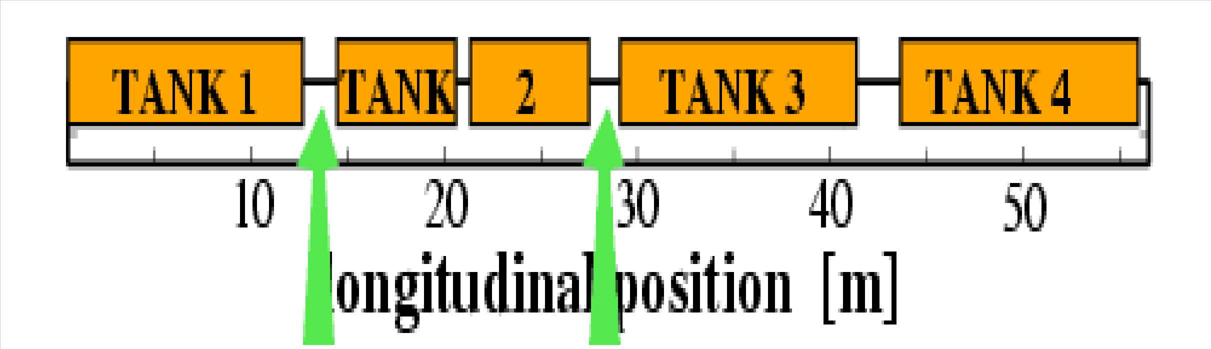


Problem: buncher cavities available?

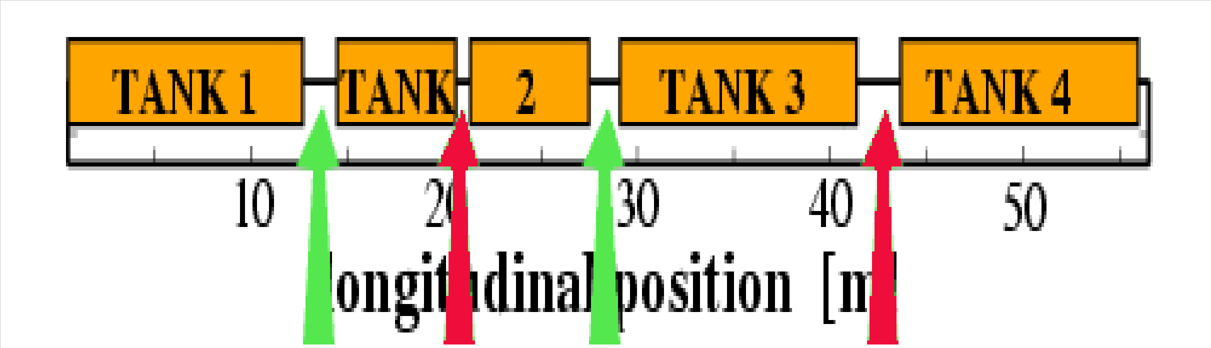


Green: Optimal positions for two buncher cavities

Problem: buncher cavities available?



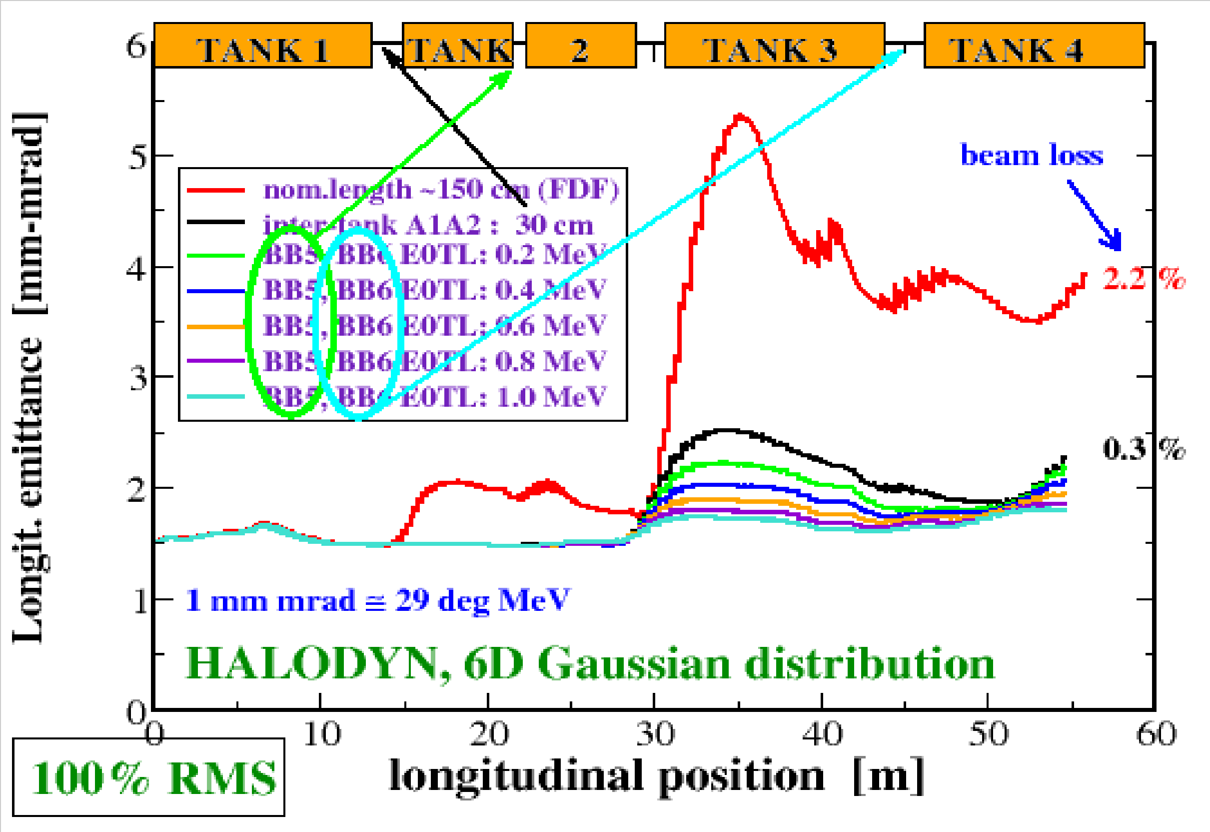
Green: Optimal positions for two buncher cavities



Red: available buncher cavities

Possible solution

Long. emittance Vs. buncher gradients with shorter A1-A2 distance



A1-A2 from 150 to 30 cm + bunchers ON at $E_0TL = 0.6$ MeV $\Rightarrow$ same results by introducing a buncher in A1-A2

Outlook

Halodyn(_MPI) ready to be installed and run onto the CNAF HPC

[picture from Wikipedia]