HELSINKI UNIVERSITY OF TECHNOLOGY
Department of Engineering
Physics and Mathematics

Mikko Voutilainen

Construction and Testing of an Electron Isolation Algorithm for Data Analysis of the CMS Experiment

Master‘s Thesis submitted in partial fulfillment of the requirements for the Degree of Master of Science in Technology.

Geneva 27.06.2003

Supervisor: Professor Martti M. Salomaa
Instructor: PhD. Katri Lassila-Perini

Acknowledgements

This thesis has been prepared at CERN in the CMS Software and Physics Project of Helsinki Institute of Physics. The whole process has taken a little over one year, with several stays at CERN totalling almost seven months and a longer break in between dedicated for studies. It has been a fruitful experience, with hard work during the cold and rainy autumn in Geneva but also inspiring lectures, excursions and experiences in an international atmosphere during my summer-studentship in the beginning of this work and later when I returned to continue it. It has been extraordinary to work in a place of excellence such as CERN, on the border of two nations, in a valley where the omnipresent mountains make everyone reach for the top.

I would like to thank Katri Lassila-Perini for very expert instruction and for setting a good example. It has been a pleasure to work with a person who is able to fluently communicate in half a dozen languages in a multinational environment and has a good track on relevant facts. I would also like to thank other colleagues at HIP and CMS for all the help and guidance I have received.

My supervisor prof. Martti M. Salomaa deserves special credit for introducing me to research work early in my studies and for providing great work opportunities. The varied special assignments made at and for the Materials Physics Laboratory have served as an excellent preparation for the thesis and given the first steps in research work.

I would like to thank my parents Keijo and Pirkko for all their support and encouragement.

Finally, I would like to thank and hug my beloved companion Tuula for all the abundance of support I've received. She has been a great help in my studies, during this work and the writing of this thesis. Tuula first introduced me to particle physics, showed an example and led the way to a Master's Thesis. We are now set to pursue the road further.

At CERN, Geneva, Switzerland
June 18$^{\text{th}}$ 2003

Mikko Voutilainen

Contents

List of Figures

List of Tables

Introduction

The work for this Master's Thesis was performed at CERN [1], the European Laboratory for Particle Physics, located at the Franco-Swiss border near Geneva. The laboratory provides the premises for an international collaboration on fundamental research in particle physics with 20 European member states and many more countries associated in many different particle physics projects. Some 7 000 scientists, in fact half of the world's particle physicists, use CERN's facilities.

The laboratory has only recently finished dismantling its 27 kilometres long Large Electron Positron Collider (LEP) and is now focusing its resources on completing the new Large Hadron Collider (LHC) that will be built in the same tunnel that used to house LEP. LHC and its two all-purpose particle detectors, CMS (the Compact Muon Solenoid) [2] and ATLAS [3] aim at finding the long-sought Higgs boson but also to find possible signatures of Supersymmetry (SUSY) or any new physics. Future research programs also include studies of CP violation at the LHCb detector [4] and the fourth detector, ALICE [5], will specialise in heavy-ion collisions and quark-gluon plasma studies.

The Higgs boson is a particle associated with the Higgs field, a hypothetical field postulated to explain the masses of the elementary particles. According to the standard model (SM), the elementary particles obtain their masses by interacting with this all-permeating field. Supersymmetry is a possible extension of the standard model and it would manifest itself as the supersymmetric counterparts of the ordinary particles. Both of these theories can be verified at the LHC which will deliver higher energies and luminosities than ever reached before in particle accelerators.

The high luminosities obtained from the LHC set very high requirements for the particle detectors. At the highest design luminosity each detector will have, on average, about 20 proton-proton collisions every 25 ns, every such bunch-crossing producing thousands of particles. Particles from the previous collision will still be flying through the detector when the next one takes places. Every beam crossing, or event, at the CMS detector will produce 16 million separate pieces of data. Combined with the 40 MHz frequency rate of beam crossings and the 1 Mbyte average event size, this corresponds to about 40 terabytes of raw data per second, far too much to be saved in memory or processed on-line by hardware.

To reduce the data flow to a manageable level, several hierarchies of triggers have to be present. Most of the events are actually uninteresting inelastic background events, while the Higgs bosons are expected to be produced only about every few seconds. First cuts (Level-1 trigger) on the data will be made by dedicated electronics based on only a few event characteristics. The high-level triggers (HLT), on the contrary, will be run on farms of off-the-shelf commercial processors. The succeeding levels will each use more event information and more time to process it. The ultimate goal of the chain of triggers is to reduce the original 40 MHz data flow down to 100 Hz that will be saved on disk and processed off-line, loosing as few interesting events as possible.

This Thesis will focus on implementing a Level-3 subtrigger for the CMS detector based on electron-isolation characteristics. This will be done using prereconstructed electron tracks and regional track reconstruction around the electron candidate. The implementation will be integrated as a part of the ORCA (Object-oriented Reconstruction for CMS Analysis) software. A major part of the Thesis will also focus on testing the feasibility of this approach with Monte Carlo simulated samples within the given constraints of computing time and the background rejection versus signal efficiency.

The outline of this Thesis is as follows:

Chapter 1 presents the basics of the standard model of particle physics. The Higgs mechanism in the standard model is elaborated in greater detail and the possible extensions of the standard model are discussed. The most important detection channels for the Higgs boson are presented.

Chapter 2 gives a brief overview of the Large Hadron Collider and the Compact Muon Solenoid experiment. The subdetectors and subsystems of the CMS experiment concerning this Thesis are discussed in more detail.

Chapter 3 illustrates the core software at the CMS experiment. The full data-handling simulation chain from data modelling the physics to detector data analysis is presented.

Chapter 4 considers triggering based on electron isolation. An implementation of an electron-track isolation algorithm is presented in detail. Results for background rejection versus signal efficiency and CPU expenditure are presented and compared to alternative solutions. A possible extension to a photon isolation trigger is discussed.

Chapter 5 summarises the results and discusses the problems encountered.

Chapter 1

The standard model

The standard model (SM) [6, 7] is the modern description of the fundamental constituents of matter and the elementary forces acting between the particles. It has evolved over a period of almost fifty years, and with the advancement of ever more powerful particle accelerators and more sensitive detectors it has become excessively scrutinised and a fundamental model in physics. One of the model's main constituents, the quantum electrodynamics (QED), has become the most accurately verified theory in modern science. Descriptive of its importance are also the numerous Nobel prizes awarded for the experimental findings leading to the model and for the theoretical development of the model [8]. In short, the standard model is our best description of the world at small distances and high energies (high velocities), as shown in Figure 1.1.

According to the standard model, all matter consists of three families of leptons and quarks and their antiparticles, listed in Table 1.1. There are four known forces acting between the particles: the strong force, the weak force, the electromagnetic force and the gravitation. Of these, gravity is too weak for its effects to be seen in any particle-physics experiment with the energies available in the foreseeable future. Combining the theoretical structures of the general relativity of gravitation and the standard model is a formidable task, and thus gravitation is currently

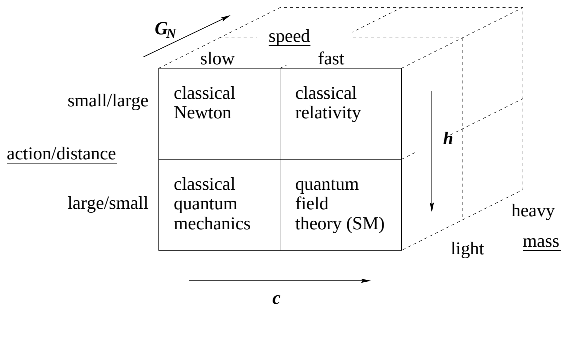


Figure 1.1: *A physicist's division of the world. The domain of the standard model lies where the changes in momentum (actions) are high, speeds approach the velocity of light and gravity is negligible. The fundamental natural constants G_N, $\hbar$ and c act as yardsticks. [9]*

excluded from the standard model. The remaining three forces are very successfully described by quantum field theories that combine quantum mechanics and special relativity. These theories are known as quantum electrodynamics that describes electromagnetic interactions, quantum chromodynamics (QCD) that describes the strong interaction between quarks, and the electroweak theory that combines QED and weak interactions. QED, QCD and the electroweak theory also fall into a special class of theories known as gauge theories, which are central in theoretical physics. In this context, the standard model can be considered to be a theory based on a gauge invariant $SU(3) \times SU(2) \times U(1)$ symmetry group where $SU(3)$ corresponds to QCD and $SU(2) \times U(1)$ to the Electroweak Theory.

In the standard model, each force is manifested as a gauge field and each gauge field in turn has gauge bosons that act as the carriers of the force, see Table 1.1. The forces then reduce to exchanges of the carrier particles. The electromagnetic force is carried by the familiar photon. The weak force is similar to the electromagnetic force but the high weight of its three gauge bosons Z^0 and $W^{\pm}$, 91.19 GeV/c^2 and 80.4 GeV/c^2 respectively [10], compared to the 1 GeV/c^2 of a proton, makes it weak compared to the electromagnetic force carried by the massless photon. The strong force on the other hand is quite different and has a total of eight gluons as force carriers. The high number of gauge bosons involved in strong interactions is explained by having not only one, as in electromagnetism, but three different "charges" or quantum states, called colours (conventionally red, green and blue). The fact that the gluons themselves carry the colour charge of strong interactions makes the theory complicated and causes the quarks to be eternally bound into white or colourless states of three differently coloured quarks or into pairs of a quark and an antiquark (carrying the anticolour). The hypothetical carrier of gravity, the spin-2 boson called graviton, is intentionally left out of Table 1.1 since nobody has been able to present a working quantum field theory of gravity to date.

The still unexplained mass of Z and $W^{\pm}$ and the remaining elementary particles requires a field, called the Higgs field [11], which will manifest itself in the form of the Higgs boson, H. The Higgs boson has not been detected yet, though, and finding this missing piece of evidence for the standard model is one of the biggest and most important challenges in contemporary particle physics. This is one of the major tasks that CERN's future Large Hadron Collider is set to solve. The Higgs boson will not, nevertheless, solve all current theoretical problems in the standard model. There are good theoretical grounds to expect new physics at the TeV energy scale that the LHC will reach. These possibilities will be discussed further in the following sections.

1.1 Elementary particles and force carriers

All ordinary matter around us is made of only four types of particles: electrons, up quarks, down quarks and electron antineutrinos. The up (u) and down (d) quarks combine into triplets to form strongly bound states which we know as the proton (uud) and the neutron (udd), collectively known as the nucleons. These

Table 1.1: *Elementary particles and force carriers in the standard model.*

Quarks +1/3	u up	c charm	t top	γ photon
-2/3	d down	s strange	b bottom	g gluons
Leptons 0	ν_e electron neutrino	ν_μ muon neutrino	ν_τ tau neutrino	$Z^0, W^\pm$ Z,W bosons
-1	e electron	μ muon	τ tau lepton	H Higgs boson
Generations	I	II	III	Gauge bosons

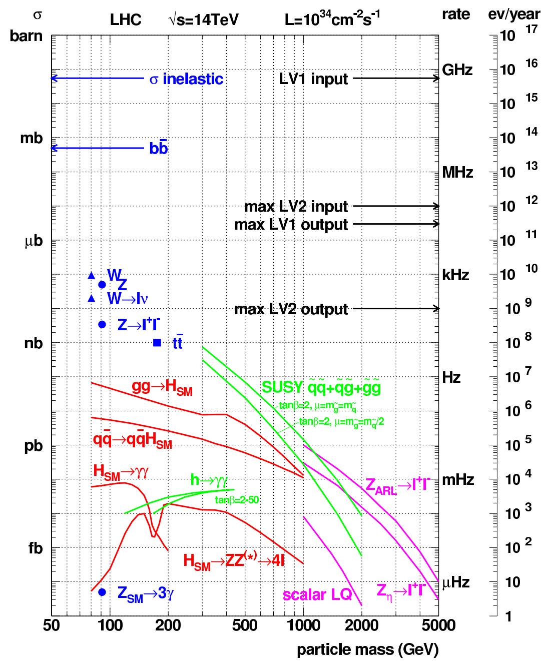
two states have the lowest possible energy of all quark combinations and are thus the only known states that are stable. To be more accurate, the proton is slightly lower in energy than the neutron (n), and free neutrons will decay into a proton (p) accompanied by an electron (e^-) and an electron antineutrino ($\bar{\nu}_e$) within a half-life of about ten minutes [10].

The protons and the neutrons combine to form nuclei in which the residual strong force between the nucleons stabilises the neutrons against decay. The additional residual strong force of the neutrons also helps to counterbalance the electrostatic repulsion between the positively charged protons in a nucleus. The negative electrons are attracted to the positively charged nuclei to form a cloud around the nuclei with well-defined, quantised energy states and wavefunctions. These bound states of electrons and nuclei are the everyday atoms, and mark the upper scale limit of the standard model as a whole. The fourth particle, the electron antineutrino, is actually the most common of the four but interacts so utterly weakly with other matter that it can hardly be seen.

The four ordinary particles already mentioned form just the tip of an iceberg. According to the standard model, all matter is composed of 12 different elementary particles. These are divided into quarks and leptons, and the quarks and leptons in turn are classified into three generations or families. Each generation has two particles of different flavour. The ordering of the elementary particles in Table 1.1 resembles Mendeleyev's periodic table of the elements in chemistry and often tempts physicists to search for an underlying structure.

With the exception of the lightest (I) generation, and possibly the muon and tau neutrinos, all other elementary particles are unstable, with lifetimes varying from the $\sim 2 \cdot 10^{-6}$ s of the muon to the $\sim 10^{-24}$ s of the top-quark. However, these unstable particles can be produced, e.g., by cosmic rays in the upper atmosphere or artificially in particle accelerators. In the LHC they are produced in the collisions of high-energy protons, or of the constituent quarks and gluons if the collision is suitably head-on.

What tells the different elementary particles apart is not only their mass, lifetime and charge but also the forces they encounter. Quarks are sensitive to all the four forces, the strong, the weak and the electromagnetic forces and the gravity.



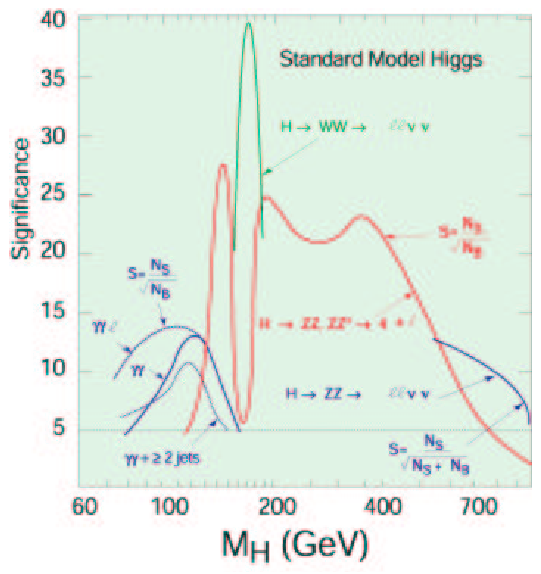


Figure 1.2: *Left: LHC background cross sections and theoretically calculated Higgs-boson production channels and cross sections as a function of the Higgs mass. [12]*

Figure 1.3: *Right: Theoretically calculated Higgs-boson decay channels and their significances as a function of the Higgs mass. [13]*

Electron flavour leptons lack sensitivity to the strong force, while the neutrinos are more sterile still. They only sense the weak force and, if they have mass, gravity, which makes them almost invisible to the rest of the matter.

We have yet to discuss the force carriers. According to QED, the electromagnetic force can be considered to be brought about through an exchange of virtual photons. Virtual particles are like their normal counterparts but have "borrowed" energy and momentum for a short period of time. This is possible according to the Heisenberg uncertainty principle which states that $\Delta E \Delta t \geq \hbar/2$. The more massive the virtual particle is, the shorter its allowed lifetime. This is what makes the weak force so much weaker than electromagnetism. At sufficiently high energies, the weak gauge bosons Z and $W^{\pm}$ have only a small rest mass compared to their kinetic energies and they behave essentially like the massless photons. This electroweak unification takes place at energy scales on the order of 1 TeV.

The carriers of the strong force, gluons, have the added complexity of being colour charged themselves. This causes them to interact between themselves and with the quarks. The strong force has also a much higher coupling constant, $\alpha_s \geq 1$, than the electromagnetic force, $\alpha \approx \frac{1}{137}$. The traditional name "constant" is actually rather misleading, since both of these parameters vary with distance and energy. The fine structure constant works essentially as a branching ratio to virtual particles and virtual particle-antiparticle pairs. This makes QCD at long distances impossible to calculate using perturbation theory and confines quarks inside the nucleons. Inside the nucleons, however, the effective strength of the force decreases, and so also α_s decreases to values much less than one, which provides the quarks with a so-called asymptotic freedom. In this special case, QCD may be used with perturbation theory as a calculational tool. The QED formalism is simpler and perturbation theory with respect to the coupling constant can be used everywhere, except extremely close to the particle.

Not only gauge bosons but also material particles can be virtual. In the world of QED and QCD, the "vacuum" is full of virtual particle-antiparticle pairs popping in and out of existence for very short periods of time. This seemingly strange behaviour of quantum fields has actually strong experimental ground. The protons, for example, do not only contain their constituent quarks but also a sea of other particles, of which the gluons and antiquarks form a major part. At every instant of time, a significant fraction (tens of percents) of a proton's momentum and energy is shared between the one down and the two up quarks and the sea of gluons and antiquarks. In fact, less than 2% of a proton's rest mass can be calculated to belong to the constituent quarks [10], the rest 98% or more is potential energy of the QCD and QED or the mass of the virtual quarks, antiquarks and gluons in the sea.

1.2 Higgs mechanism and the Higgs boson

Within the standard model, the electroweak force is broken into the weak and the electromagnetic interactions when the energy scale is below the electroweak scale. This is due to the high masses of the weak gauge bosons, W and Z. If the W and Z masses are included into the electroweak model or in the electroweak Lagrangian by hand, the model becomes inconsistent. Perturbative corrections diverge to infinity at several levels in the calculations and make the theory useless. In technical terms, such theory is called unrenormalisable [7].

To overcome these difficulties it was noted, however, that the W and Z masses can be consistently *built in* the model by introducing a new scalar field ϕ called the Higgs field. Requiring the potential to have such a form that the field is finite at the potential minimum, we can introduce spontaneous symmetry breaking into the system. This term is, however, rather misleading because the symmetry is still present, it is only hidden in the ground state, or the vacuum [7, 14].

A well-known example of spontaneous symmetry breaking is the ferromagnet that has a rotationally invariant Lagrangian. Below the critical Curie temperature T_C, all spins become aligned in the same, albeit arbitrary, direction. The ground state no longer has rotational invariance and the rotational symmetry is said to be spontaneously broken. In our case, the critical temperature T_C would correspond to the electroweak scale where symmetry breaking between the electromagnetic and weak interactions takes place.

When the Higgs field is used to break the local $SU(2) \times U(1)$ symmetry that describes the electroweak interactions, it gives rise to a new mass spectrum [14]

$$\begin{aligned}
&\text{a Higgs boson,} \quad m_H = \sqrt{2\lambda\nu^2} = \sqrt{-2\mu^2} \\
&\text{two } W \text{ bosons,} \quad m_W = \frac{1}{2}g\nu \\
&\text{a } Z \text{ boson,} \quad m_Z = \frac{1}{2}(g^2 + g'^2)^{1/2}\nu = m_W/\cos\theta_W \\
&\text{and a photon,} \quad m_\gamma = 0.
\end{aligned}$$

Here λ and μ are parameters of the Higgs field potential $V(\phi) = \mu^2\phi\phi^\dagger - \lambda(\phi^\dagger\phi)^2$, $\nu^2 = 2\phi^\dagger\phi$ and $\tan\theta_W = g'/g$. Here g and g' are constants that arise from algebra and are of no more interest here. The fact that the photon is massless should not be regarded as a prediction of the model. It is a necessary condition of electric charge conservation which in turn necessitates the choice of a neutral vacuum state. The W and Z masses were predicted and found to match the experimental values. However, the mass of the Higgs boson depends on the unknown parameter λ which appears in the potential $V(\phi)$. Nevertheless, the *existence* of the Higgs boson is a necessary condition of the model and the reason particle physicists are so eagerly searching for it. To date, it is the last missing piece of the standard model.

It might be appropriate to note one common misunderstanding that often occurs when the Higgs model is popularised. Not all mass is generated by the Higgs mechanism. Although the Higgs mechanism gives rise to the quark, lepton and Z/W masses, most of ordinary matter is composed of protons and neutrons. The u has a mass of 1.5–4.5 MeV/c^2 and the d 5–8.5 MeV/c^2 [10], so the sum of the quark masses is less than 17.5 MeV/c^2 per proton and thus only accounts for less than 2% of the nucleon mass. The rest 98%, or more, emerges from interactions within QCD and QED and it has nothing to do with the Higgs model.

The experimental difficulty of finding the Higgs boson is due to the fact that the Higgs field couples to mass. The more massive the interacting particle, the easier it is to produce a Higgs particle in collisions. Unfortunately, the experimentally easily obtainable particles are electrons and the up and down quarks. These are from the light end of the particle spectrum and couple very weakly to the Higgs field. Even with the energies and luminosities obtainable at the LHC, the Higgs cross section will be so small that these particles will only be produced every few seconds, once in roughly a billion collisions.

Although the Higgs mass itself is a free parameter of the theory, the cross sections for Higgs production and Higgs decay channels can be predicted theoretically for a given Higgs mass M_H. Figure 1.2 shows the Higgs boson cross sections and production rates at the LHC. It clearly indicates how much below the background the experimental signal lies. Obviously, this necessitates the use of a very efficient and reliable triggering system to sieve out the rare signal events from the abundant background.

Using theoretical arguments and global fits to the standard model, the Higgs boson mass can be loosely constrained to be in the range 10–1000 GeV/c^2 [10]. Direct experimental searches for a Standard-Model Higgs boson at CERN's LEP collider have excluded the mass region below 114.4 GeV/c^2 at 95% confidence

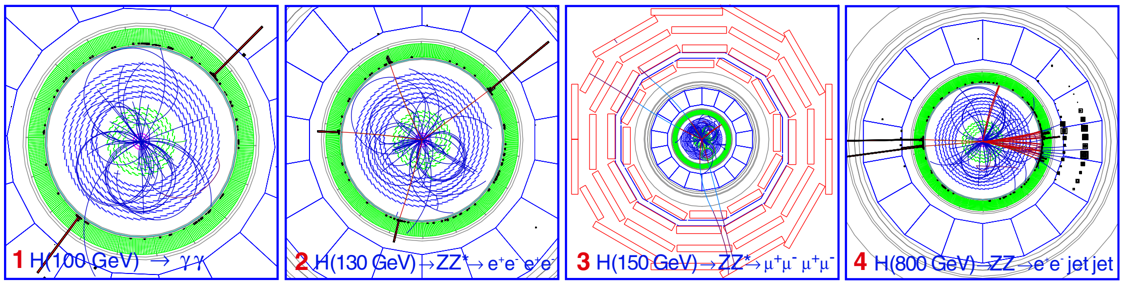

1 H(100 GeV) → γγ
2 H(130 GeV) → ZZ* → e+e- e+e-
3 H(150 GeV) → ZZ* → μ+μ- μ+μ-
4 H(800 GeV) → ZZ → e+e- jet jet

Figure 1.4: *Some of the most important Higgs boson decay signs at the CMS experiment. The concentric circles depict the tracker and the wide band ECAL (1, 2, 4). The ring of boxes represents HCAL (3, 4). The outer rings of rectangles are the muon detectors (4). [13]*

level [15]. Current data favours a light SM Higgs with a mass less than about 211 GeV/c^2 [16]. If the SM Higgs exists, it will certainly be less than 1000 GeV/c^2 and LHC will discover it with better than 5σ significance.

Figure 1.3 shows the favoured channels for finding the Higgs boson and the estimated signal significances after a few years of LHC. The relative importance of the Higgs decay channels depends on the mass of the boson, as seen in the figure. Figure 1.4 illustrates some of the corresponding experimental signals at the CMS experiment. Channels 1 and 2 rely mainly on the tracker and the electromagnetic calorimeter (ECAL). Channel 4 also uses a hadronic calorimeter (HCAL) to detect jets. Note that only channel 3 depends mainly on the other subdetectors, namely the muon detectors. The other three of the four favoured decay channels rely heavily on the tracker and ECAL. For such channels, efficient and reliable track-isolation triggering is a necessity at high luminosities.

1.2.1 Primary Higgs decay channels

The CMS experiment concentrates a lot of its effort for detecting the Higgs boson and it uses several decay channels for this purpose. Many of these channels involve an isolated electron and they are used for benchmarking the algorithm presented in this Thesis. Here we will discuss them in closer detail.

The SM Higgs boson is expected to be created mainly through gluon-gluon fusion, $gg \rightarrow H_{SM}$, or via a WW or ZZ fusion, $qq \rightarrow qqH_{SM}$ as can be seen in Figure 1.2. The Higgs boson will decay in the same way (with the same branching ratios), regardless of the process it was created in, and in the cases just mentioned the production channel does not play an important role. However, forward jets from $qq \rightarrow qqH_{SM}$ can be used for tagging, and the production channel is important if it involves an associated production of a W or Z boson, e.g. $q\bar{q} \rightarrow WH$.

The Higgs boson is most likely to decay into a massive particle, such that in the intermediate Higgs-mass region (114 GeV/c^2 $< M_H < 2M_Z$) the decay to a bottom quark pair, $H \rightarrow b\bar{b}$, will dominate, as shown in Figure 1.3. Unfortunately, this channel has a very high QCD background of $b\bar{b}$ pairs, as shown in Figure 1.2, and it is intrinsically very difficult to observe in hadron colliders. The best options are to use either the $H \rightarrow \gamma\gamma$ channel, which relies on the high energy resolution of the electromagnetic calorimeter as shown in Fig 1.4, or to trigger on associated W or Z bosons that decay to isolated leptons (l), e.g., in the process $WH \rightarrow l\nu b\bar{b}$. At the higher end of the mass region the decay to a virtual W or Z pair, $H \rightarrow WW^*$ or $H \rightarrow ZZ*$, opens a possible channel, as shown in Figure 1.4.

In the high-mass region ($2M_Z < M_H < 700$ GeV/c^2) Higgs decay to real W and Z pairs dominates. These decay in essentially the same way as virtual pairs, e.g. $H \rightarrow ZZ \rightarrow llll$. In the ultimate high mass region ($M_H > 700$ GeV/c^2), complementary pair decay channels $ZZ \rightarrow lljj$ and $WW \rightarrow l\nu jj$, see Figure 1.4, become important as the production cross section of other channels falls as a function of the Higgs mass.

What concerns electron isolation triggering, the main signals containing isolated electrons involve $W \rightarrow e\nu$ and $Z \rightarrow e^+e^-$ decays. These were chosen for benchmarking the signal efficiency. For an extension to photon isolation, the most important channel is $H \rightarrow \gamma\gamma$ that was chosen for benchmarking the diphoton efficiency. Specialised channel-specific background cuts will be done during off-line analysis and are not considered here. Therefore, both electron isolation and photon isolation jet rejections were tested against the QCD background generated in five bins and properly weighted.

1.3 Extensions to the standard model

Precision measurements of the standard model have shown that the theory is accurate in many of its predictions to less than one part in a billion at energies lower than about 100 GeV. It has withstood experimental testing for over 20 years and no experimental evidence against it exists, yet we know it is incomplete. It leaves too many questions unanswered to the "ultimate theory". Some of the current hot questions are the origin of the mass of the elementary particles (a possible solution could be the Higgs field), the Grand Unification of the forces at high energies, matter-antimatter asymmetry (CP violation) and the nature of the non-baryonic dark matter in the universe (stable supersymmetric particles have been proposed). Neither the standard model has anything to say about gravity, the dominant force in the universe over cosmic distances. Einstein's theory of general relativity is a good theory but it does not tell us how gravity behaves at the very small distances and large energy densities that took place during the Big Bang. We simply have no valid theory of quantum gravity that would tie together the cosmic distance scales and the quantum world of elementary particles.

One possible extension to the standard model is supersymmetry (SUSY) [17]. In this model, every particle has a supersymmetric counterpart, listed in Table 1.2. The Supersymmetric partner of a fermion is a boson and vice versa. The relative masses of the particles are also inverted in supersymmetry. These supersymmetric particles would need to be heavier than ordinary matter to have escaped detection in particle-physics experiments performed thus far. The lightest supersymmetric

Table 1.2: *Supersymmetric partners of elementary particles and force carriers. In the minimal supersymmetric standard model there are five Higgs bosons, $h, H^0, A, H^\pm$, other models may have more.*

Squarks +1/3	$\tilde{u}$	$\tilde{c}$	$\tilde{t}$ stop	$\tilde{\gamma}$ photino
-2/3	$\tilde{d}$	$\tilde{s}$	$\tilde{b}$ sbottom	$\tilde{g}$ gluinos
Sleptons 0	$\tilde{\nu}_e$ electron neutralino	$\tilde{\nu}_\mu$ muon neutralino	$\tilde{\nu}_\tau$ tau neutralino	$\tilde{Z}^0, \tilde{W}^\pm$ Zino,Winos
-1	$\tilde{e}$ selectron	$\tilde{\mu}$ smuon	$\tilde{\tau}$ stau	$h^0, H^0, A^0, H^\pm$ Higgsinos
	I	II	III	Gauginos

particle, a linear combination of the particles in Table 1.2 and often dubbed χ, would be stable and a suitable WIMP (Weakly Interacting Massive Particle) candidate that could explain the origin of dark matter in the Universe. It would also have to interact very weakly with ordinary matter.

Within SUSY, there are not just one but at least five different Higgs bosons. The minimal supersymmetric standard model (MSSM) [18], called so because it adds the minimal amount of free parameters to the SM, contains five Higgs bosons, h^0, H^0, A^0 and $H^\pm$. The lightest of these, h, would be detectable at the energies attainable by LHC possibly along with other light SUSY particles. The other four MSSM Higgs bosons may also be detectable at the LHC, depending on the parameters of the model. The masses of the lightest SUSY particles, if they exist, are expected to be in the TeV scale. Their signatures in particle detectors would be very distinct and require in many cases an isolated electron. Examples of such channels are $A, H \rightarrow \tilde{\chi}\tilde{\chi} \rightarrow 4l^-_{\text{isol}} + E_t^{\text{miss}}$, where $l \in \{e, \mu, \tau\}$, or $A, H \rightarrow \tau\tau \rightarrow e\nu_e\nu_\tau + \mu\nu_\mu\nu_\tau$. Many more channels include isolated electrons among the end products of a supersymmetric cascade.

Chapter 2

Large Hadron Collider and the CMS experiment

When the Large Hadron Collider (LHC) [19] will start its operation in 2007, it will resume the mantle of the world's largest collider from the Fermi National Laboratory's (Fermilab) Tevatron collider. With a centre-of-mass energy of $\sqrt{s} =$ 14 TeV, it will surpass the existing colliders by almost an order of magnitude. The nominal luminosity $\mathcal{L} = 10^{34}$ cm^{-2}s^{-1} will be some 100 times higher than in the existing colliders, adding new technical challenges to detection triggering and data acquisition. These advances supply the LHC great discovery potential, and the whole particle physics community is eagerly awaiting its start-off.

The LHC will be a host to four major experiments, CMS, ATLAS, ALICE and LHCb. The first two are large multi-purpose detectors with a diverse experimental program. Their main goals are finding the Higgs boson, measuring its properties and confirming or disproving Supersymmetry. ALICE, A Large Ion Collider Experiment, focuses on studying quark-gluon plasma that will be formed in heavy-ion collisions. The smallest of the experiments, LHCb, is dedicated to studying CP violation in B-meson (b-quark bound with a lighter quark) decays.

2.1 Accelerator technology

The LHC is a 27-kilometre long proton-proton synchrotron accelerator ring. Because the charges of the two proton beams have the same sign, the accelerator has two parallel beam pipes running in the opposite directions. The bending dipole magnets are designed such that the magnetic field changes sign for the different beam pipes. The bending magnetic field that is needed to keep the 7 TeV protons on track is 8.36 T, made possible by superconducting magnets. This adds new technical challenges, since the magnets have to be cooled down to cryogenic temperatures (1.9 K) and kept there despite beam power-loss side-effects, such as 3.6 kW synchrotron radiation.

Each proton beam consists of 2835 proton bunches, and each bunch in turn has 10^{11} protons. The high beam current, 0.54 A, and the resulting energy losses present a particular challenge for the cryogenics. The high density and energy of the colliding beams also cause other side-effects, such as the beam-beam effect, wake-fields, photo-electron clouds, and even chaotic behaviour, that will act to destabilise the beam and degrade its quality. During the beam life-time of about 10 hours the particles make 400 million revolutions around the machine. The LHC will have a large number of focusing quadrupoles, careful design of beam pipe structure and choice of materials, and high-quality collimators to address these problems, to remove unstable particles and to retain a high beam quality over the whole beam life-time [19].

2.2 The Compact Muon Solenoid experiment

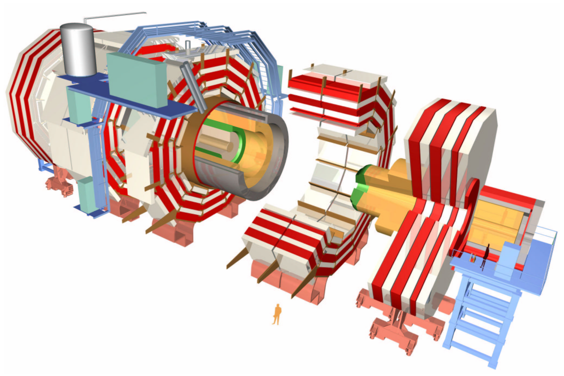


Figure 2.1: *A 3D model of the CMS detector. One of the barrel muon rings and the endcap are moved out and the detector sliced to reveal the inner structure containing the hadron calorimeter, electromagnetic calorimeter and the tracker inside the solenoid magnet. [20]*

CMS, the Compact Muon Solenoid, is not the biggest, but rather the heaviest experiment in the LHC. The name Compact comes from its special feature of having the muon chambers inside the return yoke of the magnet system. It is based on a single superconducting solenoid, the largest superconducting magnet ever built, that is capable of creating a magnetic field of 4 Tesla. Such an intense field is needed to curve the trajectories of very-near-to-the-speed-of-light particles enough to accurately measure their momentum. The outer parts of the CMS are large dedicated muon chambers that will accurately measure the muons' energy and momentum. Muons are especially important in some Higgs decay channels, such as $H \rightarrow ZZ^* \rightarrow \mu^+\mu^-\mu^+\mu^-$.

The CMS detector has been designed to detect cleanly the diverse signatures from new physics by identifying and precisely measuring muons, electrons and photons over a large energy range and at high luminosity [2]. The most important projects include studying Higgs physics, Supersymmetry, B-physics (CP violation), and heavy-ion collisions that will create quark-gluon plasma. To reach these goals, CMS has been equipped with highly sensitive silicon pixel and silicon strip detectors for tracking, an accurate electromagnetic calorimeter (ECAL), a brass-scintillator hadronic calorimeter (HCAL) and efficient muon gas chambers. Figure 2.2 shows the radial subdetector composition of CMS.

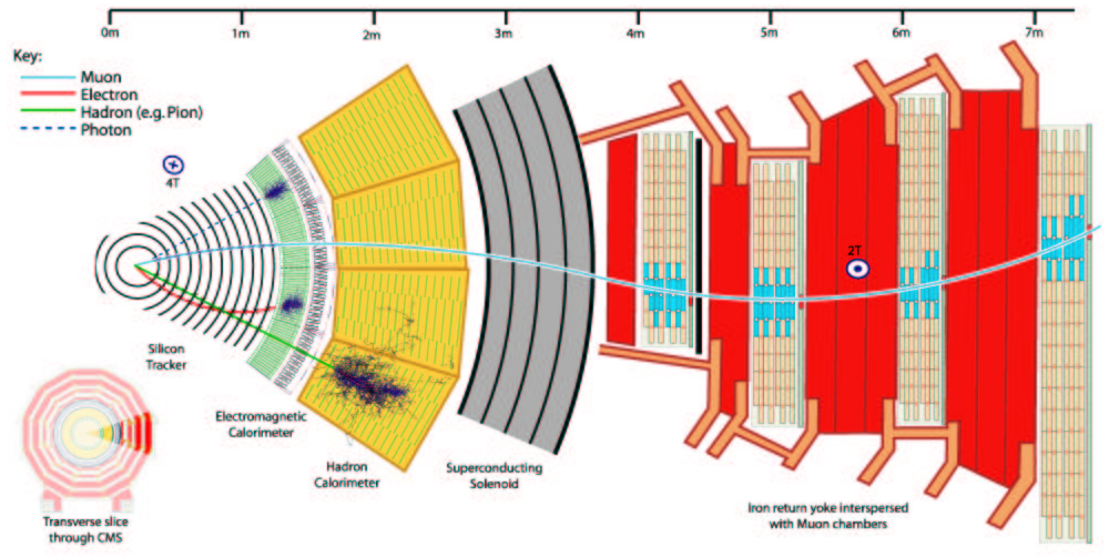


Figure 2.2: *The CMS subdetectors from inside out: silicon pixel detector (inner tracker, 3 layers), silicon strip detector (outer tracker, 10 layers), electromagnetic calorimeter (ECAL), hadronic calorimeter (HCAL), superconducting solenoid magnet, iron return yoke and muon chambers. [20]*

For studying electron isolation, the relevant subdetectors are the tracker and ECAL. These will be elaborated in more depth in the following two subsections.

2.2.1 CMS tracker

The CMS tracker is composed of two types of detectors [21, 22]: the three innermost barrel layers and two innermost end disks are made of silicon pixel detectors and the ten outer layers in the barrel and twelve vertical layers in the endcaps are made of silicon strip detectors, as shown in Figure 2.3. The detectors were chosen for their high spatial resolution but also for their radiation tolerance. The radiation levels at the LHC are unprecedented and detectors near the beam crossing are required to withstand unusually high radiation doses.

The purpose of the inner tracker is to provide high precision points near the interaction vertex. The design goal is to achieve an impact parameter resolution at high p_t of order 20 μm in the transverse plane and 100 μm in the z-direction [2]. This will allow efficient secondary vertex finding and b-tagging. The high spatial-resolution points are also essential in pattern recognition at high luminosities. Having a few well-defined track measurements near the interaction point will significantly reduce combinatorial ambiguities. Table 2.1 summarises some essential parameters of the pixel detectors.

Table 2.1: *Summary of some pixel-detector parameters. [21]*

Detector type	layer	position [cm]	pixel size [μm]	resolution [μm]
barrel pixel	1	4.1–4.5 (r)	150 ($r\phi$)$\times$ 150 (z)	10–15 μm
	2	7.0–7.4 (r)	as above	as above
	3	10.7–11.2 (r)	as above	as above
endcap pixel	1	$\pm$32.5 (z)	150 (r)$\times$150($r\phi$)	15–20 μm
	2	$\pm$46.5 (z)	as above	as above

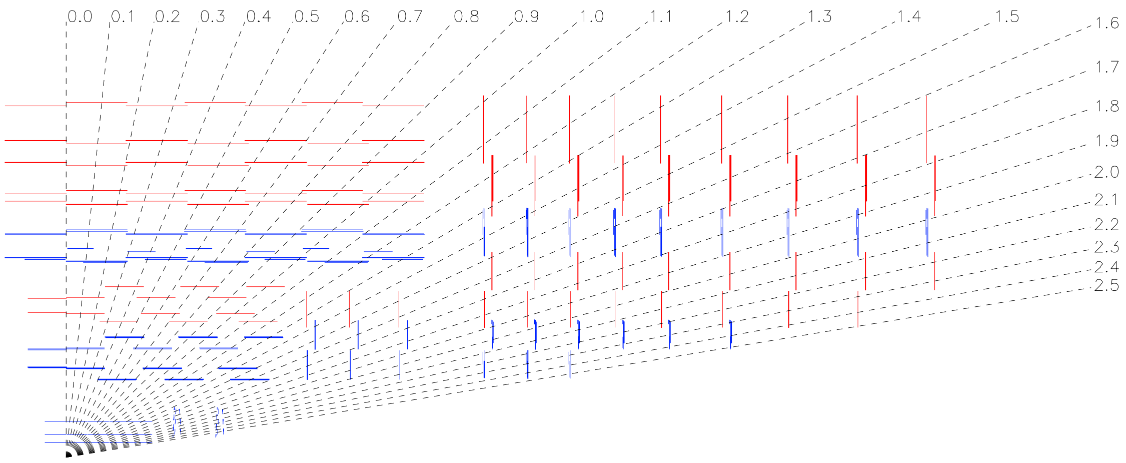


Figure 2.3: *An $r - z$ view of a quadrant of the CMS tracker. The three small barrel layers and two disks in the lower left corner are silicon-pixel detectors. The outer layers are made of silicon strip detectors. The dashed lines show different pseudorapidities. [23]*

The outer tracker layers are needed for high momentum resolution and track finding. Careful pattern-recognition studies have shown that tracks can be reconstructed with high efficiency, provided that 10–12 points per track are recorded [2]. The design goal of the central tracking system is to reconstruct isolated high p_t tracks with an efficiency of better than 95%, and high p_t tracks inside jets with an efficiency of better than 90% over the pseudorapidity range $|\eta| < 2.6$, defined as $\eta = -\ln(\tan\theta/2)$, where θ is the polar angle. The momentum resolution required for isolated charged leptons in the central rapidity region is $\Delta p/p_t \approx 0.1 p_t$ (p_t in TeV). This will allow assignment of lepton charge up to $p_t \approx 2$ TeV [2].

2.2.2 CMS electromagnetic calorimeter

CMS electromagnetic calorimeter (ECAL) [24] uses lead tungstate (PbWO$_4$) crystals to detect the incoming particles. Lead tungstate is a scintillating material and, as such, a fully active medium. The energy resolution is usually parametrised [24] as

$$\left(\frac{\sigma}{E}\right)^2 = \left(\frac{a}{\sqrt{E}}\right)^2 + \left(\frac{\sigma_n}{E}\right)^2 + (c)^2 \qquad (E \text{ in GeV}) \qquad (2.1)$$

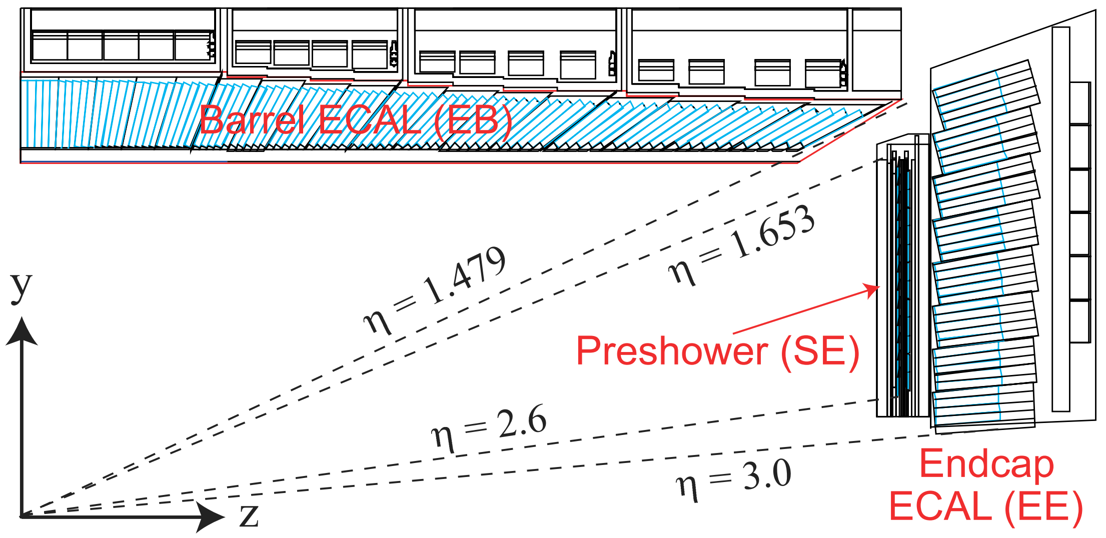


Figure 2.4: *Transverse section of ECAL, as described in GEANT3/CMSIM (version CMS125). [23]*

where a is the stochastic term, σ_n the energy equivalent of noise and c a constant term. For CMS ECAL, these contributions are [24] for a 5×5 crystal array $a = 2.7\%$ for the barrel ($0 \leq |\eta| \leq 1.479$) and $a = 5.7\%$ for the endcap ($1.479 \leq |\eta| \leq 2.61$), $\sigma_n = 155$ MeV for the barrel and $\sigma_n = 770$ MeV (E_T =205 MeV) for the endcap at low luminosity, and $c = 0.55\%$. For high luminosity the corresponding values for σ_n are 210 MeV and 915 MeV (E_T =245 MeV). Figure 2.4 shows the ECAL geometry.

Compared to sampling calorimeters, active calorimeters have a lower stochastic term a (Eq. 2.1) and thus usually yield a better energy resolution. The physics process that imposes the strictest performance requirements on the ECAL is the $H \rightarrow \gamma\gamma$ channel. Thus the diphoton mass resolution is used as a benchmark for the ECAL performance. With the above-mentioned resolutions, this would be roughly 1%.

The crystals form a grid in η, ϕ with a granularity of $\Delta\eta = \Delta\phi = 0.0174$ [24]. The trigger division of $\Delta\eta \times \Delta\phi = 0.087 \times 0.087$ corresponds to a group of 5×5 crystals. In both projections crystals make an angle of 3° with lines from the interaction point to avoid projective cracks.

2.2.3 CMS data acquisition

With the different subdetectors combined, the CMS has some 16 million separate channels producing data for each bunch crossing [2, 25], as shown in Figure 2.5. The tracker alone accounts for 10 million of these. At the crossing rate of 40 MHz this produces a staggering amount of raw data per second. To synchronise the data from different subdetectors and to allow the electronics time to respond, the data is first fed to a 3 Gigacell buffer, some 200 cells per channel. The Level-1 trigger processing logic then considers local information from front-end electronics concerning trigger primitives, such as photons, electrons, muons and jets, as well as global sums of E_T and missing E_T. The global trigger decision is given after a fixed latency of 3.2 μs. This step reduces the event rate to 100 kHz.

At the first level of processing, the raw data is compressed using techniques such as zero suppression. This reduces the event size to 1 Megabyte. In the Event Builder, separate pieces of information from the subdetectors are collected into one event and then sent for the on-line processor farm for Level-2 decision. The farm will consist of high-performance commercial processors. The workload will be distributed such that a single CPU only processes one event at a time. The work on the event continues on the same processor farm through Level-2.5 and the final Level-3 decisions, and the event is then saved to archive for later off-line processing.

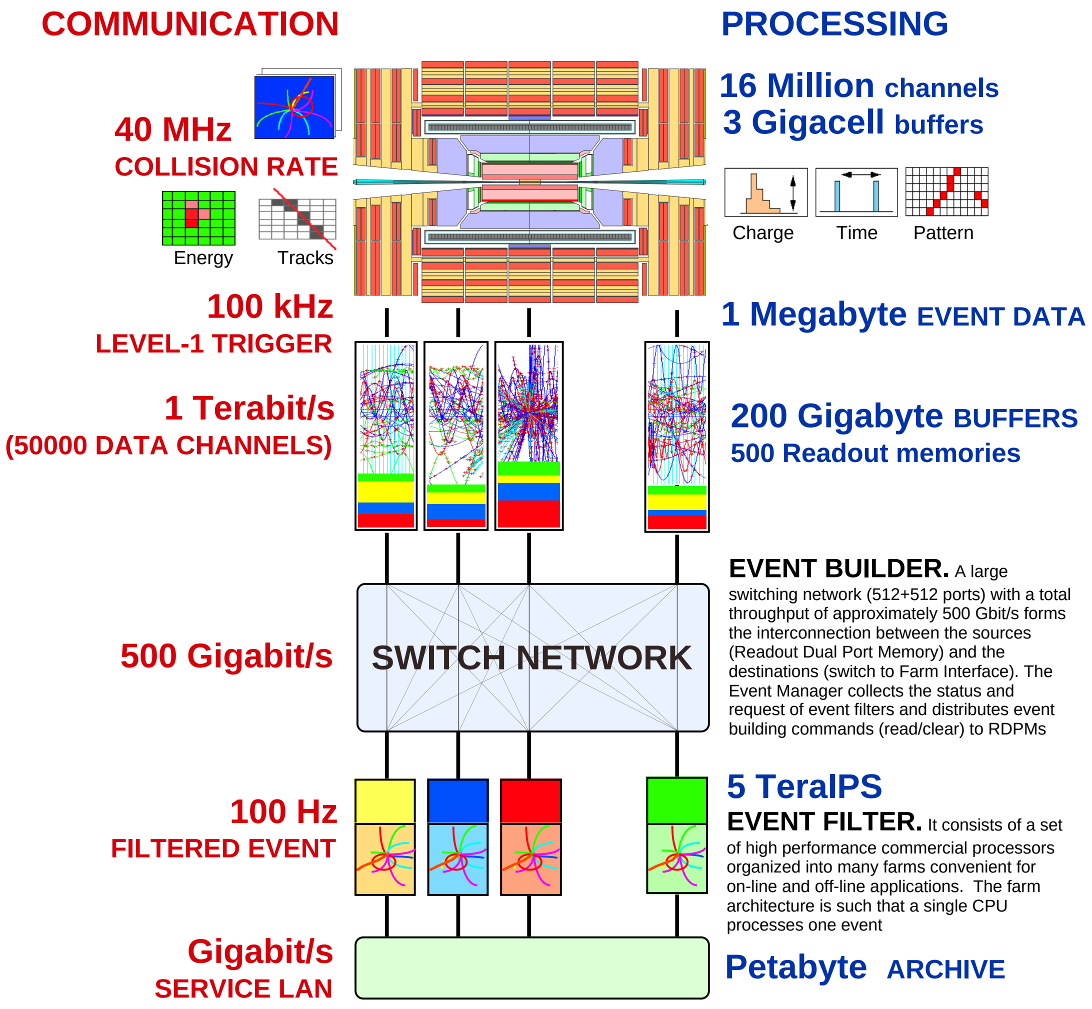

COMMUNICATION
PROCESSING
40 MHz
COLLISION RATE
Energy
Tracks
16 Million channels
3 Gigacell buffers
Charge
Time
Pattern
100 kHz
LEVEL-1 TRIGGER
1 Megabyte EVENT DATA
1 Terabit/s
(50000 DATA CHANNELS)
200 Gigabyte BUFFERS
500 Readout memories
500 Gigabit/s
SWITCH NETWORK
EVENT BUILDER. A large switching network (512+512 ports) with a total throughput of approximately 500 Gbit/s forms the interconnection between the sources (Readout Dual Port Memory) and the destinations (switch to Farm Interface). The Event Manager collects the status and request of event filters and distributes event building commands (read/clear) to RDPMs
100 Hz
FILTERED EVENT
5 TeraIPS
EVENT FILTER. It consists of a set of high performance commercial processors organized into many farms convenient for on-line and off-line applications. The farm architecture is such that a single CPU processes one event
Gigabit/s
SERVICE LAN
Petabyte ARCHIVE

Figure 2.5: *CMS data acquisition system.* *IPS: Instructions Per Second, LAN: Local Area Network*

The processor farm uses software tailored for the CMS. The Level-2 decision only considers ECAL data but Level-2.5 and Level-3 triggers request the rest of full crossing data for processing. Building the high level triggers (HLT) on software that can be run on commercial processors is a very cost-efficient and flexible solution. The trigger software can be relatively easily upgraded and the same algorithms and programs can be used both for on-line and off-line processing and Monte Carlo simulations. In fact, the CMS computing project is based on the assumption that computing power of commercial processors increases exponentially according to the well-known Moore's law. Computer performance per money spent doubles roughly every 1.2 years [26], so $\sim$32 times more computing power is expected to be available for the same price at the time of the commissioning of the LHC in 2007 than was available when the building started in 2001.

Chapter 3

CMS software

In a modern high-energy physics experiment, most of the data analysis is made using computers. When the data from the LHC will be processed, only a very small fraction of the events selected will ever be directly scrutinised by a human using visualisation programs such as IGUANA [27]. Many other tasks, such as monitoring, triggering, calibration, etc., are also more-or-less fully computerised. To handle these data amounts efficiently and reliably, many software tools have been developed for physics simulation, detector simulation, event triggering and reconstruction, and data-analysis, to name just a few. This chapter will give a brief overview of the most common software tools in use at the CMS and CERN, and especially of those related to this Thesis. The interrelations of the software packages are visualised in Figure 3.1.

Until the LHC start in 2007, no real physics data will be produced for the CMS data analysis. Nevertheless, physicists are hard at work perfecting their physics reconstruction and data-analysis tools using simulated data. Let us first give a brief summary of the full simulation chain before embarking on a more elaborate description of the different software packages.

A hypothetical simulation chain starts with the generation of physics events in an event generator, such as Pythia [28]. These events are preselected according to loose Level-1 cuts to reduce the amount of unnecessary computations needed in the subsequent steps. The preselected physics events are run through detector simulation in CMSIM/GEANT3 [29, 30]. This transforms the particles and tracks of the physics events into observables, such as tracker hits and energy deposited in calorimeter crystals, simulates detector efficiencies, electronics noise and statistical fluctuations, and gives output similar to the one that will once be obtained from the real CMS detector. To complete the detector simulation, ORCA [31] is used to digitise the analog data to resemble the output of CMS electronics. The raw simulated data is then processed by simulated Level-1 trigger and several software triggers in ORCA. ORCA will also take care of the final event reconstruction and event tagging. The requested event data will be written on disk in compact form as an ntuple for physics analysis. In the last stage, physicists will analyse the data using tools, such as PAW [32].

It should be noted that many of the programs mentioned above are about to be revised in the near future. Due to the increasing complexity of the programs, CMS has adopted an Object Oriented (OO) approach [26]. This means that all new software is produced in object oriented C++ and old programs coded in Fortran have largely been rewritten or are in the process of being rewritten in object oriented C++. The switch to C++ makes code reuse and share easier, and enables the use of software development methodologies that meet the CMS goals for software reliability.

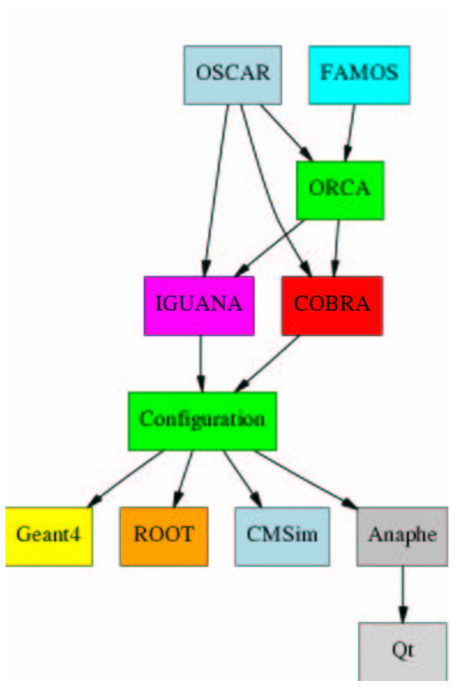


Figure 3.1: *Major CMS software packages and their interrelations. OSCAR, CMSim (soon obsolete) and FAMOS are programs used for detector simulation. ORCA is used for software triggering, event reconstruction and data-analysis. COBRA contains common routines for reconstruction, analysis and simulation shared by ORCA and OSCAR. IGUANA is used for visualisation. Geant4 is a toolkit for the simulation of the passage of particles through matter. ROOT is an object oriented framework for large scale data-analysis. CMSim is also still used as a detector description database for OSCAR but will soon become obsolete. Anaphe (formerly LHC++) and Qt contain libraries for visualisation and data-analysis.*

3.1 Physics and detector simulation

Several programs for physics event generation have been developed, Pythia [28] probably being the most extensive and widely used at CMS. It is a part of the so-called "Lund Monte Carlo" family, software authored by the Lund University theory group. Pythia generates Monte Carlo simulations of collisions at high energies between elementary particles, such as e^+, e^-, p and $\bar{p}$. It contains theory and models for a number of physics aspects, including hard and soft interactions, parton distributions, initial and final state parton showers, etc. The current production version, Pythia 6, is based on Fortran but will be replaced by C++ based Pythia 7.

The detector simulation for Monte Carlo production is currently done by the Fortran based CMSIM/GEANT3 combination. These will be replaced by OSCAR [33], Object oriented Simulation for CMS Analysis and Reconstruction, and Geant4 [34] in the near future. Both will follow the CMS practise of using C++. Functionality from the physics viewpoint will, however, be minimally affected.

The purpose of the detector simulation is to transform the physics events into detector measurements. The detector efficiencies, noise levels and measurement uncertainties will be simulated in detail. The output of the detector simulation will closely resemble the (analog) data the true experiment will produce. Detailed detector simulation is time consuming and in some cases rough approximations for detector response will be sufficient. FAMOS [35], FAst MOnte Carlo Simulation, can be used for such purposes.

To complete the simulation, ORCA is used to digitise the analog data to mimic real CMS electronics output. ORCA can also be used to simulate the Level-1 trigger electronics to filter simulated events. The software triggers at Level-2 to Level-3 are also implemented in ORCA.

3.2 ORCA, Object Oriented Reconstruction for CMS Analysis

One of the most important software packages for CMS is ORCA, Object oriented Reconstruction for CMS Analysis. As the name suggests, ORCA takes care of the event reconstruction but it also implements the software triggers that will be run on a CPU farm when CMS is operational. Other smaller tasks include digitisation of analog detector simulation data from OSCAR, CMSIM or FAMOS.

Following CMS computing guidelines, ORCA is fully based on the Object Oriented approach. The current internal structure of ORCA is shown in Figure 3.2. For some parts of its functionality, ORCA also relies on other software packages. Most important of these is COBRA [36], Coherent Object-oriented Base for Reconstruction, Analysis and simulation. It provides services common to both ORCA and OSCAR. ORCA visualisation is handled through IGUANA, Interactive Graphics for User ANAlysis. The detector geometry description will in the future be handled mainly through DDD [37], Detector Description Database, that replaces the current CMSIM version CMS125 description of the CMS detector. Object persistency, i.e., object storage to memory, relies currently on commercial Objectivity databases but will be integrated with CERN authored ROOT [38] as of ORCA version 7.

This Thesis will be mostly involved with the ElectronPhoton and TrackerReco packages within ORCA. ElectronPhoton provides tools and algorithms for the reconstruction of electromagnetic clusters, including schemes for isolation and π^0

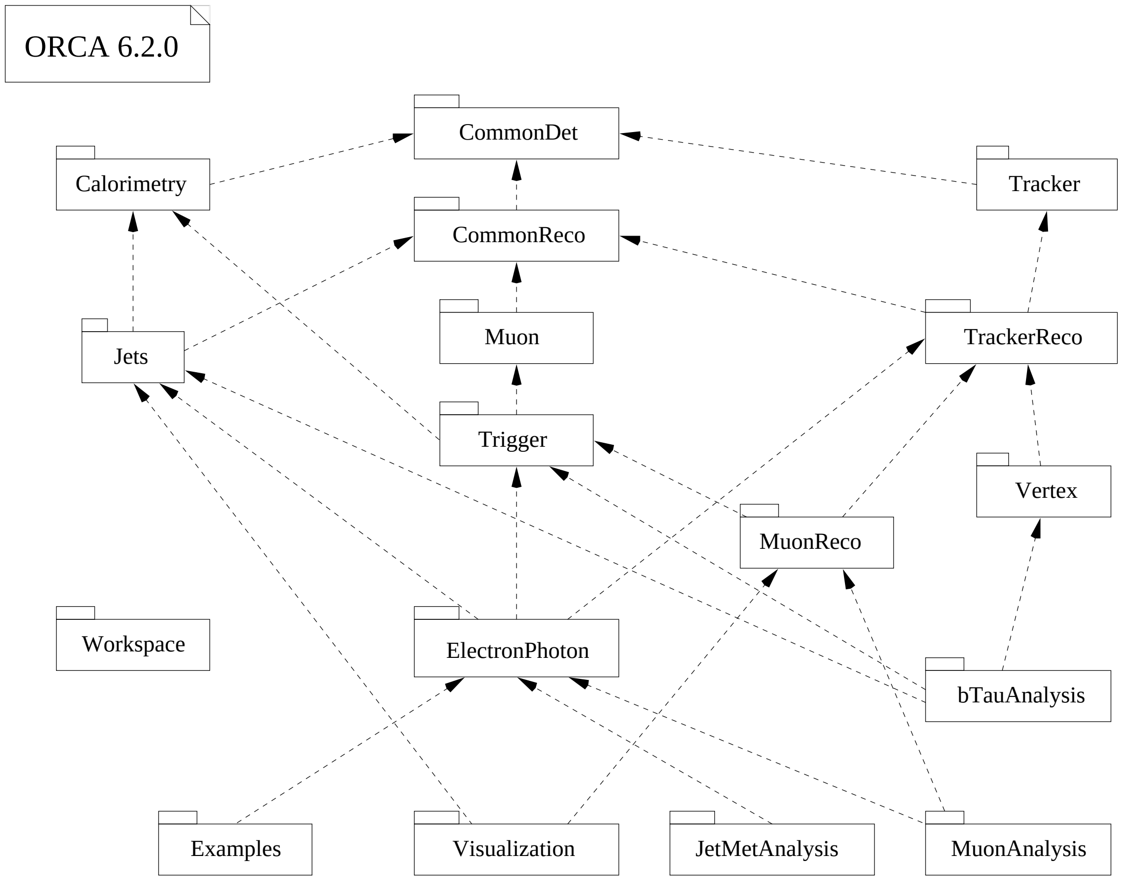


Figure 3.2: *Package interrelations inside ORCA. [39]*

rejection. Search for pixel seeds associated with an ECAL cluster is included. ElectronPhoton also provides Level-1 and high level trigger (HLT) selection facilities. Level-1 selection uses indirectly the Trigger package that simulates a Level-1 trigger. The high level triggers reside in ElectronPhoton.

TrackerReco provides framework for tracker reconstruction. It contains especially the tools needed for performing regional reconstruction of the tracks inside the isolation cone. TrackerReco also provides the PixelVertexReconstruction and PixelVertexFinder classes that are essential for finding photon vertex candidates.

Chapter 4

Electron isolation selection cut

The CMS experiment will record two proton bunches crossing every 25 ns, i.e., at the rate of 40 MHz. At the nominal design luminosity, often also referred to as high luminosity, of 10^{34} cm^{-2}s^{-1}, each bunch crossing will yield on average about 20 inelastic proton-proton interactions and hundreds of new particles. Data from 16 million individual detector channels is read out for triggered events, giving an event size of about 1 MB. To be able to store the incoming data, the rate has to be reduced from 40 MHz down to about 100 Hz using a series of triggers and HLT selection cuts, as explained in subsection 2.2.3. This is also schematically shown in Figure 4.1. The reduced data stream can then be stored and later processed off-line.

The majority of all collisions produce jets by hard scattering processes. These QCD jet events form the main background that we need to remove. Rarer are events where massive particles are produced. Weak gauge bosons, Z and $W^{\pm}$, for example, will be produced in thousands every second, while heavier top quarks will appear a few times a second and Higgs bosons would be created about once every few seconds at high luminosity. Some decays, like a Higgs boson decaying into four leptons, would be detected only about once a day.

While it is certainly necessary to have high luminosity in order to produce such rare events as Higgs bosons at a tolerable rate, the hardware costs and the available technology set a limit to the amount of data that can be fully or even partially processed. Level-1 triggers will reduce the data rate from 40 MHz to about 100 kHz using a subset of the event data, mostly transverse energy E_t from the different calorimeters. The Level-1 trigger system [2] processes the trigger primitives generated by front-end electronics with the processing logic in electronics barracks. The subsequent High-Level Trigger (HLT) will be provided by an on-line processor farm.

In order to save CPU time, the HLT is divided into several steps. There are no sharp boundaries as all steps are run on the same processor farm, but the selection cuts are conventionally named Level-2, Level-2.5 and Level-3, according to the order in which they are applied. The Level-2 selection again considers a subset of data. After a positive Level-2 decision, the remainder of the full crossing data is requested for further processing by subsequent levels. The final decision to keep the event is taken at Level-3. At this stage the event rate has been reduced to about 100 Hz and will be stored to memory for off-line processing.

The selection cut discussed in this Thesis is just one of several selection cuts. It uses information from the ECAL and, in the latest step, from the tracker, which means that it will act mainly on electron and photon streams but also on tracks left by some other charged particles for isolation purposes. Other selection cuts will also look for muon signals in the muon tracker, for jets in the hadronic calorimeter (HCAL) or a mixture of all these signals for global triggering.

4.1 Physical and technical background

After the Level-1 rough energy measurement in 5×5 crystal trigger towers, the energy is measured more precisely at Level-2 in more complete and flexible ECAL super-clusters, also accounting for the energy lost by Bremsstrahlung. Higher E_T cuts are then applied on the newly calculated values. The main background passing the Level-2 ECAL selection cut are π^0s, few of which contain a matching track.

At Level-2.5, the ECAL super-clusters are matched to the tracker hits by extrapolating an electron (or a positron) trajectory back to the z-axis and searching for compatible hits in the inner silicon pixel layers. If such hits are found, the event is deemed to have an electron candidate and will pass Level-2.5. Otherwise, higher E_T cuts are used to separate a photon stream. The rest of the events are rejected. This will remove most π^0s and the main background will then be $\pi^0/\pi^\pm$ overlap, $b/c \to e$ decays and jets containing a high-p_t π^0 [25] that decays into an e^+e^--pair and a γ (1.2 % of all π^0 decays [10]). All of these backgrounds are characterised by particles in jets. They simply mimic the properties of an isolated high-p_T particle closely enough to pass the ECAL isolation trigger at Level-1, the E_T cuts at Level-2 and either pixel matching or photon candidate E_T cuts at Level-2.5.

Level-3 refers to the selection that involves the reconstruction of full tracks in the tracker. A pixel track, or a pixel line, is a partially reconstructed track consisting of 2–3 matching pixel-detector hits. A fully reconstructed track, on the other hand, contains also matching hits in the silicon strip detectors. With correct statistical weighting, these additional hits improve the primary vertex and momentum resolution and also reduce the possibility of ghost tracks. This terminology will be needed when we discuss different electron isolation algorithm implementations.

Isolation cuts are vital elements of electron and photon selection. It is possible to implement these using pixel line and/or regional track reconstruction, both of which are discussed in this Thesis. The algorithm used for track isolation is first developed for electron tracks and then extended for use with the photon candidates.

The most important signals we are interested in will contain a heavy gauge boson decay $W \to e\nu$, $Z \to e^+e^-$ or a Higgs boson decay $H \to \gamma\gamma$. The second channel is also related to the Higgs boson through the decays $H \to Z + Z$, $H \to Z + Z^*$, where Z^* denotes a virtual Z. All three channels are characterised by an isolated electron or a photon. These issues were discussed in more detail in section 1.2.

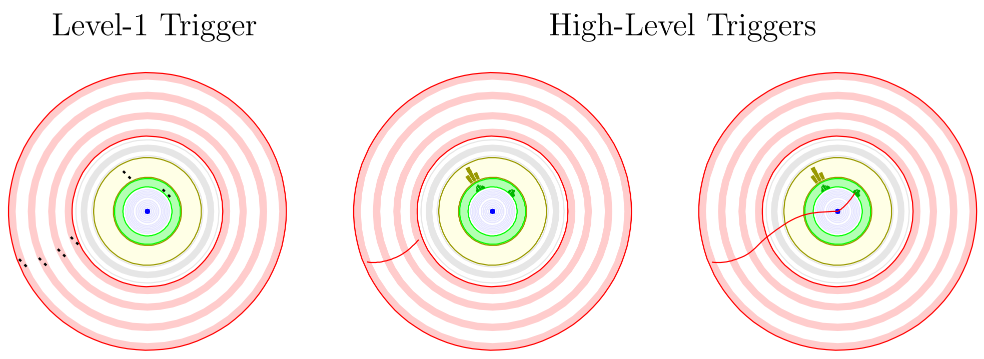


Figure 4.1: *Schematic view of the CMS multilevel trigger system. The Level-1 trigger gives a coarse and fast estimate of the measured quantity, and the subsequent higher-level trigger steps bring in more data and refine the measurement. [40]*

4.2 Implementation of the electron isolation algorithm

The algorithm for cutting on isolated electrons is mostly based on ORCA classes already available. Regional track reconstructions around the electron track seeds are used to limit the CPU resources needed for determining electron isolation. The tracks that are reconstructed outside the given (η-ϕ) isolation cone and beyond the allowed values for track transverse momentum p_t, distance from the vertex in the transverse plane Δr and in the beam-axis Δz are then removed. This often loses as much as 2/3 of the reconstructed tracks in the case of jets but improves efficiency and predictability of the algorithm. Appendix A shows the source code in the form in which it is included in the ORCA.

The general outline of the algorithm is quite simple because it is based on high-level abstractions. The analysis code receives an event in the form of a G3EventProxy, which is then filtered through the Level-1, Level-2 and Level-2.5 selection cuts.

```
void testTrackIsolation::userAnalysis(G3EventProxy *event) {
  EgammaL1Selection *l1sel = EgammaL1Selection::instance();
  EgammaL2Selection *l2sel = EgammaL2Selection::instance();

  int l1response = l1sel->Select(); // positive int means pass
  int l2response = l2sel->Select(); // positive int means pass
  int l25response = l2sel->Select25(); // positive int means pass

  if (l25responce <= 0) return;
```

Level-1 simulates the front-end electronics trigger, Level-2 does selection cuts using more detailed information from ECAL and Level-2.5 matches ECAL super-clusters to pixel detector hits. Next, the G3EventProxy *event* is asked for electron candidates in the form of RecEvent “EPTracks”. If an ElectronCandidate class is later implemented it could be used instead.

```
RecItr<TTrack> tracks(event->recEvent(),''EPTracks''); // Iterator
```

These electron candidates are requested for their momenta at the vertex,

```
RecTrack rt = RecTrack(*tracks); // cast TTrack to RecTrack
rt.momentumAtVertex();
```

and for the global coordinate of the vertex.

```
GlobalPoint vtx(0,0,rt.impactPointState().globalPosition().z());
```

Note that at this stage the vertex is actually not yet fully reconstructed; therefore, instead of asking for the candidates vertex (rt.vertexPosition()) we have to content ourselves with the z-coordinate of the impactPointState. The impactPointState is defined as the state at closest approach to vertex in the transverse plane. The transverse coordinates are both explicitly set to zero, which is a very good approximation for practically all primary vertices.

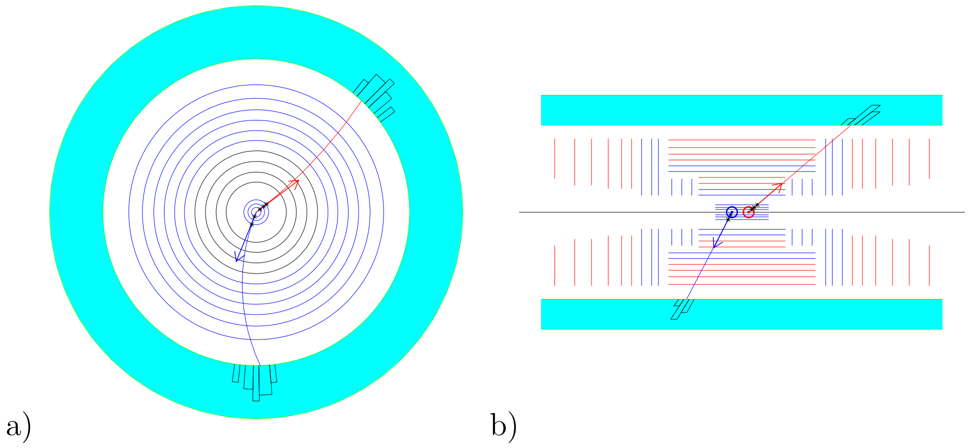


Figure 4.2: *Regional reconstruction algorithm in EgammaTrackIsolation is given the direction of an electron tracks (RecTrack) momentum* **(a,b)** *and the z-coordinate of the vertex* **(b)** *as input. Projections are from the transverse plane* **(a)** *and plane parallel to the beam line* **(b)**.

The regional track reconstruction algorithm is given the direction of the electron candidates momentum and the position of the primary vertex, as shown in Figure 4.2,

```
RectangularEtaPhiTrackingRegion
  rectRegion(mom, vtx, ptMin, deltaR, deltaZ, etaMargin, phiMargin);
theRegionFactory =
  new TrackingRegionSimpleFactory<RectangularEtaPhiTrackingRegion>;
theRegionFactory->setRegion(rectRegion);
theRegional =
  new SeedGeneratorFromPixel<NewSeedGenerator>(theRegionFactory)
theTrackFinder = new CombinatorialTrackFinder();
theTrackFinder->setSeedGenerator(theRegional);
```

and asked to find all tracks with p_t above a certain fixed threshold $PtMin$ and inside a rectangular (η-ϕ) isolation region ($etaMargin, phiMargin$) around the seed track. Figure 4.3 shows the large-scale shape of the isolation region.

```
vector<RecTrack> regionalTracks; // for storing output of reco
theTrackFinder->reco(regionalTracks);
```

The Δr ($deltaR$) and Δz ($deltaZ$) constants are needed to allow some tolerance in the reconstruction of the vertex position. With higher values of Δr, electrons from secondary b-decay vertices will also be included.

To have the final decision on the electron isolation, the reconstructed tracks are iterated to verify that they indeed do fit inside an isolation cone of a fixed radius $R = \sqrt{\eta^2 + \phi^2}$ around the electron track (the corners of the original rectangular isolation region will be cut!) with proper p_t, Δr and Δz. Note that although all the reconstructed tracks in Figure 4.3 have several hits inside the isolation region, only the original electron tracks actually fulfil all of the constraints for ΔR, Δz, Δr and minimum p_t. Typically up to 2/3 of the reconstructed tracks might be discarded at the iteration step.

```
for (vector<RecTrack>::const_iterator it=regionalTracks.begin();
     it!=regionalTracks.end(); it++) { Check constraints }
```

If the isolation region contains only one track or no tracks at all (the original seed track may sometimes be lost outside the region if the isolation cone is too narrow or the p_t threshold is too high), the electron candidate is deemed isolated. In some channels, such as $Z \to e^+e^-$, there are more than one electron candidate. In such cases the event is isolated only if both the electrons are isolated.

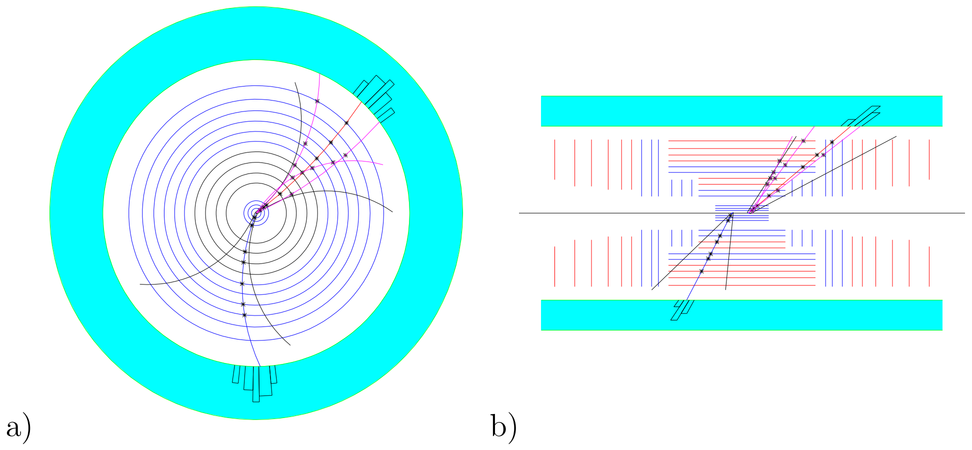


Figure 4.3: *EgammaTrackIsolation reconstructs tracks from hits inside a rectangular isolation region around the electron track. In the transverse plane* **(a)** *the black region boundaries are a combination of $\Delta\phi$ and minimum p_t. Here $\Delta\phi$ determines the opening at the vertex, minimum p_t the amount of curvature. In the plane parallel the beam line* **(b)** *the boundaries are determined by $\Delta\eta$. The Δr and Δz widen the boundaries slightly but this is not visible in the scale of the figures.*

Table 4.1: *Statistics for the Monte Carlo samples used in this study.*

DatasetName	Generated	Simulated	X-Sect (mb)	Used	Selected
eg02_jets_pt1520	2604714	100000	1.94	99914	289
eg02_jets_pt2030	4255222	400000	0.816	201837	2125
eg02_jets_pt3050	837723	200000	0.195	199781	8102
eg02_jets_pt50100	226792	120000	0.0287	119898	12243
eg02_jets_pt100up	57402	50000	0.00171	49962	6435
eg02_Wenu	29864	20000	0.0000201	19749*	19749*
eg02_Zee	40826	20000	0.00000186	19984*	19984*
eg02_H115_gg	20000	20000	0.653E-10	19739*	19739*

**Number of valid events*

Collection ID/Name
1132: eg_L1_1034PU_Tk_g125_CERN eg02_jets_pt1520
1133: eg_L1_1034PU_Tk_g125_CERN eg02_jets_pt2030
1183: eg_L1_1034PU610_Tk_g125_CERN eg02_jets_pt3050
1185: eg_L1_1034PU610_Tk_g125_CERN eg02_jets_pt50100
1184: eg_L1_1034PU610_Tk_g125_CERN eg02_jets_pt100up
1153: eg_1034PUeg602_Tk_g125_BRIS eg02_Wenu
1040: eg_1034PUeg602_Tk_g125_CERN eg02_Zee
1024: eg_1034PUeg602_Tk_g125_CERN eg02_H115_gg

4.3 Results

4.3.1 Tools used in this study

The algorithm for electron isolation was implemented using ORCA 6.2.3 and ORCA 6.3.0. ORCA 6.2.3 is currently the official production version, and it was used for processing the Monte Carlo samples described below. For CPU analysis, the later version ORCA 6.3.0 was used because it includes an easy-to-use timer. The differences between the two are minor and do not affect the results. A later version of the electron isolation code has been updated to work with the development versions of ORCA 7 but this has not been used in this study.

To analyse the performance of the algorithm, samples from the 2002 Monte Carlo production were used [41]. Pythia 6.1 [42] was used to generate events for the signal and the background. The GEANT-based full detector simulation package CMSIM [29] was used to simulate the detector response. For background, a loose Level-1 filtered jet sample was chosen. This sample is divided into five p_t bins (15–20, 20–30, 30–50, 50–100 and over 100), according to the generated transverse momentum of the interaction. For the signal, samples $W \to e\nu$, $Z \to e^+e^-$ and $H \to \gamma\gamma$ for high luminosity (10^{34} cm^{-2}s^{-1}) and with pile-up (PU) from CMS production [43] were used. The statistics of these bins are listed in Table 4.1.

For performance studies, the samples were first processed with the ORCA Level-1, Level-2 and Level-2.5 selections and the electron (photon) isolation code, EgammaTrackIsolation, introduced in this study. The parameters (vertex coordinate and momentum vector, among others) of the tracks found inside the isolation cone were stored in ntuples for final cuts in PAW. The optimisation of isolation parameters was made using selection tests provided in the PAW syntax. PAW not only allowed easy histogramming of the results but more importantly, selecting events with varying cuts from an ntuple is several thousands of times faster than rerunning the selection chain in ORCA. The results, however, remain unchanged when looser cuts are used in ORCA than in PAW.

The necessary cut parameters for Level-1, Level-2, Level-2.5 and EgammaTrackIsolation (Table 4.2), among other input parameters for other ORCA classes, can be set using the *.orcarc* input file. This greatly reduces the amount of recompilations needed and simplifies the procedure of changing runtime parameters for the program. Some ORCA related parameters are also set in the *examples.env* file.

For Level-1 selection, thresholds for transverse energies in ECAL trigger towers are set for single E_T^{sng} and double E_T^{dbl} electron-photon streams. ECAL isolation is also required from both single and double streams, except in the relaxed trigger, where no isolation is required above the E_T threshold. Level-2 thresholds are set for transverse energies in the ECAL super-clusters E_T^{sng} and E_T^{dbl}. The Level-1 trigger towers are associated to the Level-2 super-clusters in the ECAL fiducial region of $|\eta| < 2.5$, with $1.4442 < |\eta| < 1.5660$ excluded. At Level-2.5, asymmetric search regions $\phi_{min,e}$ and $\phi_{max,e}$ in the transverse plane are used for finding pixel detector hits corresponding to the ECAL super-clusters. These search regions are set separately for electrons and positrons.

4.3.2 Efficiency and jet rejection factor

The efficiency and background (jet) rejection factor are the standard criteria when considering the performance of the selection cut algorithm for different signal channels. The efficiency ϵ is defined as the fraction of signal events passing the filter,

$$\epsilon = \frac{n_{\text{passed,signal}}}{\text{n}_{\text{total,signal}}}, \tag{4.1}$$

and is often given in percents. Here $n_{\text{total,signal}}$ refers to the number of signal events passing Level-2.5 selection that contain exactly one electron candidate. In roughly 20% of the $W \to e\nu$ events passing Level-2.5 the electron candidate is lost. The jet rejection factor ρ is defined as the inverse of efficiency for background events, i.e.,

$$\rho = \frac{n_{\text{total,background}}}{n_{\text{passed,background}}}. \tag{4.2}$$

Here, accordingly, $n_{\text{total,background}}$ refers to the number of background events that pass L2.5 and contain an electron candidate. Roughly 1/3 of the background events passing L2.5 do not contain any electron tracks. For both the signal and background events, cuts shown in Table 4.2 have been used at the Level-1, Level-2

Table 4.2: *Cuts used for Level-1, Level-2 and Level-2.5 selections in the .orcarc.*

Selection	Cut variable	Threshold	*.orcarc* flag
Level-1	E_T^{sng}	28 GeV	L1Sele:SingleEMEtThr=28.
	$E_T^{dbl,1}$	14 GeV	L1Sele:DoubleEMEtThr1=14.
	$E_T^{dbl,2}$	14 GeV	L1Sele:DoubleEMEtThr2=14.
	e_1 isolated	true	L1Sele:SingleIso=1
	e_1, e_2 isolated	true	L1Sele:DoubleIso=1
Relaxed	$E_T^{dbl,1}$	19 GeV	L1Sele:DoubleEMEtRelaxedThr1=19.0
Level-1*	$E_T^{dbl,2}$	19 GeV	L1Sele:DoubleEMEtRelaxedThr2=19.0
Level-2	E_T^{sng}	31.0 GeV	L2Sele:sngEt1=31.0
	$E_T^{dbl,1}$	16.9 GeV	L2Sele:dblEt1=16.9
	$E_T^{dbl,2}$	16.9 GeV	L2Sele:dblEt2=16.9
	–	true	L1L2Assoc:InFiducial=1
Level-2.5	ϕ_{min,e^-}	-0.025	EPHitMatch:ePhiMin1 = -0.025
	ϕ_{max,e^-}	0.015	EPHitMatch:ePhiMax1 = 0.015
	ϕ_{min,e^+}	-0.015	EPHitMatch:pPhiMin1 = -0.015
	ϕ_{max,e^+}	0.025	EPHitMatch:pPhiMax1 = 0.025

* *The Level-1 isolation cuts are not used for relaxed trigger.*

and Level-2.5 selections. These are the current estimates of the final on-line cuts [44].

To obtain correct overall results, the rejection factors of the five different p_t bins have to be weighted correctly. For the signal, no weights are needed because all events are from the same collection. The first step for background is to weight the events in the p_t bins according to the cross section, also rescaling the number of the generated events by the number of events used in the Level-1 selection

$$w_{ev} = X\text{-sect}/(N_{\text{generated}} N_{\text{used}}/N_{\text{simulated}}). \tag{4.3}$$

The right-hand-side values are listed in Table 4.1. The difference between the number of generated events and the number of simulated events arises from the Pythia preselection which is applied [45]. To also account for the true number of the events used in this study, N_{L1}, slightly smaller than the available number N_{selected} due to technical difficulties, we multiply by the inverse of the fraction of events used, $N_{\text{selected}}/N_{L1}$, to obtain the event weight w at Level-1

$$w = w_{ev} \frac{N_{\text{selected}}}{N_{L1}}. \tag{4.4}$$

The bin weight K at Level-2.5 for events having a single electron track is obtained by multiplying the event weight w_{ev} by the number of events in the bin N_{selected} and then weighting by the fraction of Level-1 events passing Level-2.5 and having a single electron track $N_{L2.5}$

$$K = w_{ev} N_{\text{selected}} \frac{N_{L2.5}}{N_{L1}}. \tag{4.5}$$

Table 4.3: *Background and signal rejection at different trigger levels. Boldface numbers indicate the values to which the isolated events in this study are normalised.*

DatasetName	Used	Level-1	Level-2	Level-2.5	+0e	+1e	+2e
eg02_jets_pt1520	289	18	1	0	0	**0**	0
eg02_jets_pt2030	2130*	252	37	1	1	**0**	0
eg02_jets_pt3050	8098	1719	698	45	22	**21**	2
eg02_jets_pt50100	12091	2564	1668	144	55	**79**	8
eg02_jets_pt100up	5316	1293	1001	111	36	**54**	20
eg02_Wenu	8000	3832	3065	2819	540	**2249**	29
eg02_Zee	6000	5385	5097	4830	351	2080	**2362**

**Five events duplicated, a known bug in the sample*

The overall efficiency can then be obtained in two equivalent ways

$$\epsilon = \frac{\sum_i w_i S_i}{\sum_i w_i T_i} \tag{4.6}$$

$$\epsilon = \sum\nolimits_i K_i \frac{S_i}{T_i} \Big/ \sum\nolimits_j K_j = \sum\nolimits_i K_i' \epsilon_i, \tag{4.7}$$

where $T_i = N_{L2.5,i}$ is the number of events passing Level-2.5 with a single track, $S_i = \epsilon_i N_{L2.5,i}$ is the number of isolated events and the primed weights K_i' are normalised to one, $\sum_i K_i' = 1$. The equivalence of the two forms is easily seen by substituting $K_i = w_i T_i$ into Eq. (4.7).

The statistical error of the number of isolated events in each bin is estimated as $\sigma_i = \sqrt{S_i}$, where S_i are assumed to be Poisson distributed. The overall error estimate for efficiency $|\Delta\epsilon|$ is then calculated as a root-mean-squared of the weighted $\Delta\epsilon_i$

$$|\Delta\epsilon| = \sqrt{\sum\nolimits_i (K_i'\Delta\epsilon_i)^2} = \sqrt{\sum\nolimits_i (K_i'\sqrt{S_i}/T_i)^2} = \sqrt{\sum\nolimits_i (K_i'\epsilon_i)^2/J_i}, \tag{4.8}$$

where we have $J_i = \max\{S_i, 1\}$ to avoid division by zero. The error in rejection factor $1/\epsilon$ is simply

$$|\Delta(1/\epsilon)| \approx |\Delta\epsilon|/\epsilon^2. \tag{4.9}$$

For the signal the error estimate is similar but the number of isolated events S_i in the equations above is replaced by its complement $1 - S_i$ and the efficiency ϵ by $(1 - \epsilon)$ to avoid unrealistically high error limits when ϵ is close to one.

The efficiency versus jet rejection factor can be tuned by varying the isolation cone size and by changing the $p_{t,min}$ threshold for the reconstructed tracks. The main source of inefficiency for the signal events, such as $W \to e\nu$ and $Z \to e^+e^-$, are the pile-up events which may contribute additional tracks to the isolation region. Reducing the isolation cone size or increasing the p_t thresholds will increase efficiency but simultaneously decrease the jet-rejection factor. The tracks in pile-up events tend to be softer than the tracks in events mimicking an isolated electron. This allows jet-rejection factors close to 4.5 to be obtained with a comfortable single-electron efficiency of 95%, or 3 for 98%, see Figure 4.4 and Table 4.5.

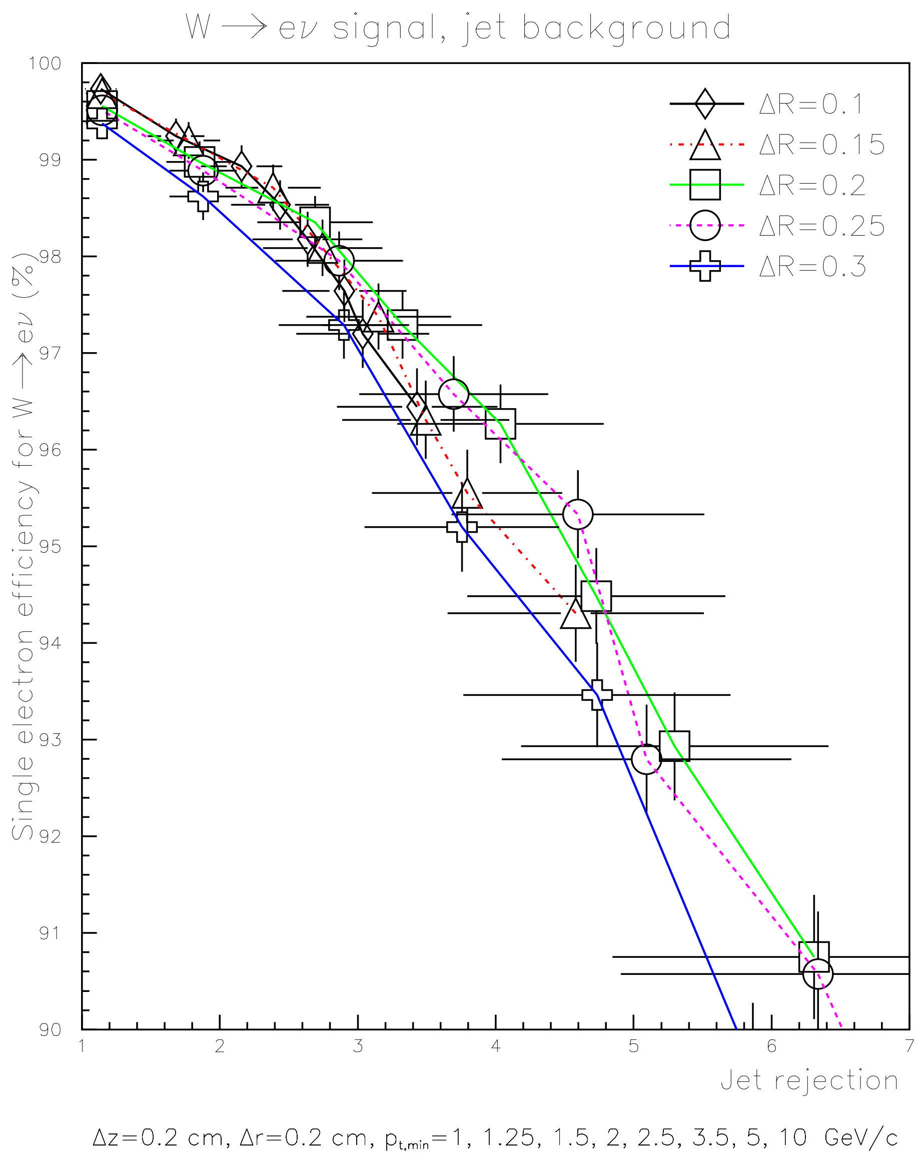


Figure 4.4: *Single-electron efficiency versus background rejection for $W \to e\nu$ signals compared to weighted jet background bins. In this plot, only events containing exactly one electron candidate have been included. However, the results will not change appreciably if events containing more than one electron are included ($\sim 1\%$ of signal, $\sim 18\%$ of jets). Black error bars indicate 1σ statistical error.*

In the case that the event contains two electron candidates, we can estimate the selection cut performance by a simple calculation from the single-electron performance. Given a single-electron efficiency P_e and assuming the second electron to have the same efficiency (which is, strictly speaking, not the case) we can calculate the double-electron efficiency P_{2e} as

$$P_{2e} = P_e^2 \tag{4.10}$$

Expressing signal efficiency as $P_s = (1 - \epsilon_s), \quad \epsilon_s << 1$ and the background efficiency $P_b = \epsilon_b, \quad \epsilon_b << 1$, we may write the estimates

$$\begin{aligned} P_{2s} &= (1-\epsilon_s)^2 \approx 1 - 2\epsilon_s, \quad \text{and} \qquad (4.11) \\ P_{2b} &= \epsilon_b^2. \qquad (4.12) \end{aligned}$$

For single-electron efficiencies 95% and 98% and respective jet-rejection factors 4.5 and 3 the double-electron channel would have roughly 90% and 96% efficiencies for the jet-rejection factors 20 and 9, respectively. The actual efficiency versus estimated jet-rejection performance is shown in Figure 4.6.

The above analysis breaks down if the second electron does not have the same efficiency as the first one. The two electrons cannot be considered completely independent since they have passed through a series of filters which are different for the double electrons than for single electrons. In general, the p_t distribution of the second electron will be somewhat lower than that of the first electron. However, if the efficiency is not strongly dependent of the p_t of the electron's track, the two efficiencies will be roughly the same and the results of the analysis hold. Figure 4.5 shows the efficiency of background electron candidates, or jet efficiency ϵ_{jet}, as a function of the track p_t.

For analysing the efficiency versus jet rejection for the $Z \rightarrow e^+e^-$ channel, a sample of 6000 signal events were chosen. This translates to 2362 events passing Level-2.5 and having double-electron candidates, a fair sample with regard to statistics. The background analysis turned out to be quite a lot trickier. The statistics of the full loose Level-1 jet sample at Level-2.5 with single-electron candidates, 187 events in total, is mediocre at best. The corresponding statistics with double-electron candidates is unacceptably low with only 30 background events passing the long chain of selection cuts.

Therefore, to gain a rudimentary estimate of the algorithmic performance for $Z \rightarrow e^+e^-$, the double-electron jets were replaced with single electrons and the efficiency per bin was calculated using the above analysis, with $P_{2e} = P_e^2$. It was expected that the double-electron jets might be relatively more numerous in the higher p_t bins, an assumption the results back up, so despite the low statistics the bin weights were calculated using the distribution of double-electron events in the bins 3050, 50100 and 100up.

At fixed p_t and ΔR cuts, the signal efficiency seems to follow the above analysis within the error bars, which lends credence to the assumption that the same might work for the background, too. Also, the change in the bin weights seems to modify the rejection factors only modestly.

4.3.3 CPU time analysis

Although the time requirements after Level-2.5 are less demanding than before and the available average CPU time per event processed is estimated to be on the order

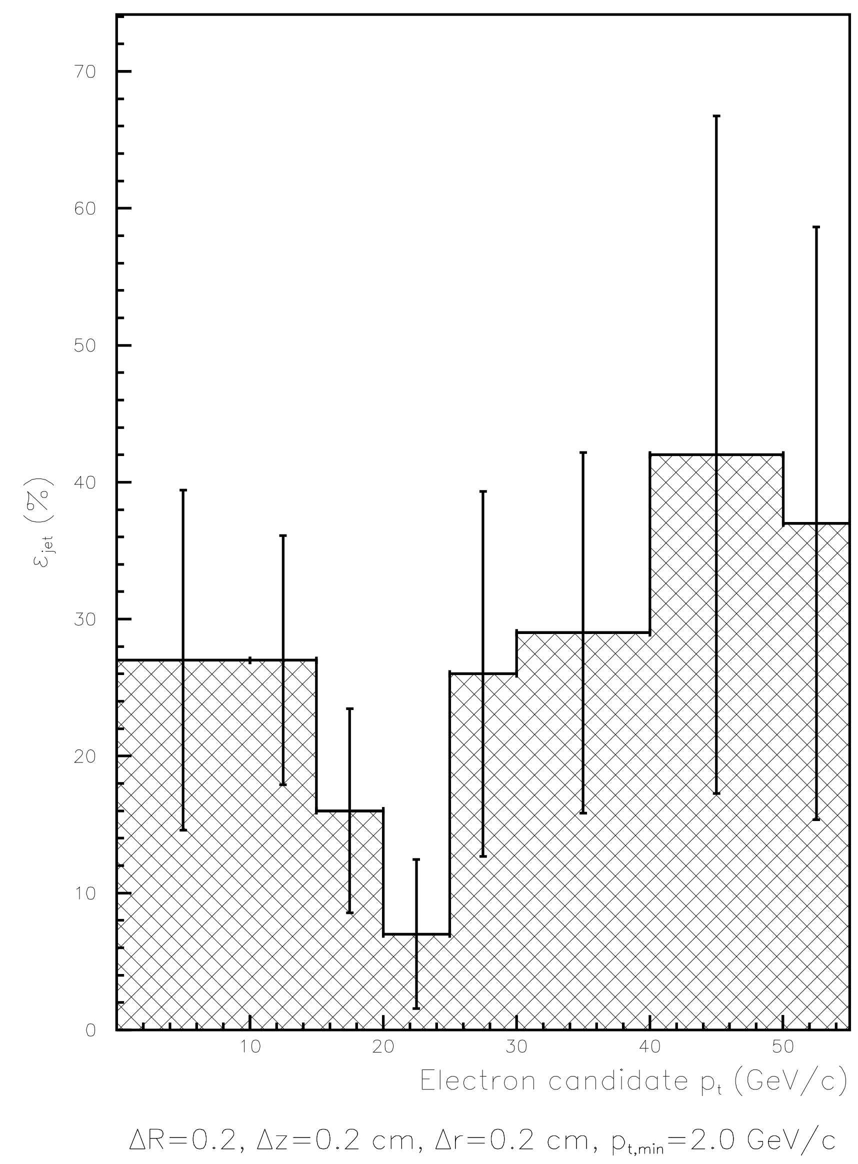


Figure 4.5: *Jet efficiency ϵ_{jet} as a function of p_t. Results were obtained calculating the efficiency for the same events passing Level-2.5 and having a single-electron candidate as were used in the $W \to e\nu$ analysis. These events were then binned according to the electron candidate p_t. Events from different p_t bins were given the same weight. Note that the bins are not of equal width but were adjusted to allow reasonable statistics. The last bin contains all events with $p_t > 50$ GeV/c.*

of one second, the CPU consumption is nevertheless an important benchmark for the algorithm. The latest versions of ORCA (from ORCA6.3.0 onwards) include an automatic CPU timer which is used by calling the constructor

```
TimeMe myTimer(``Reconstructing_EgammaIsolationTracks'');
```

The destructor automatically stops the timer when the program exits the scope

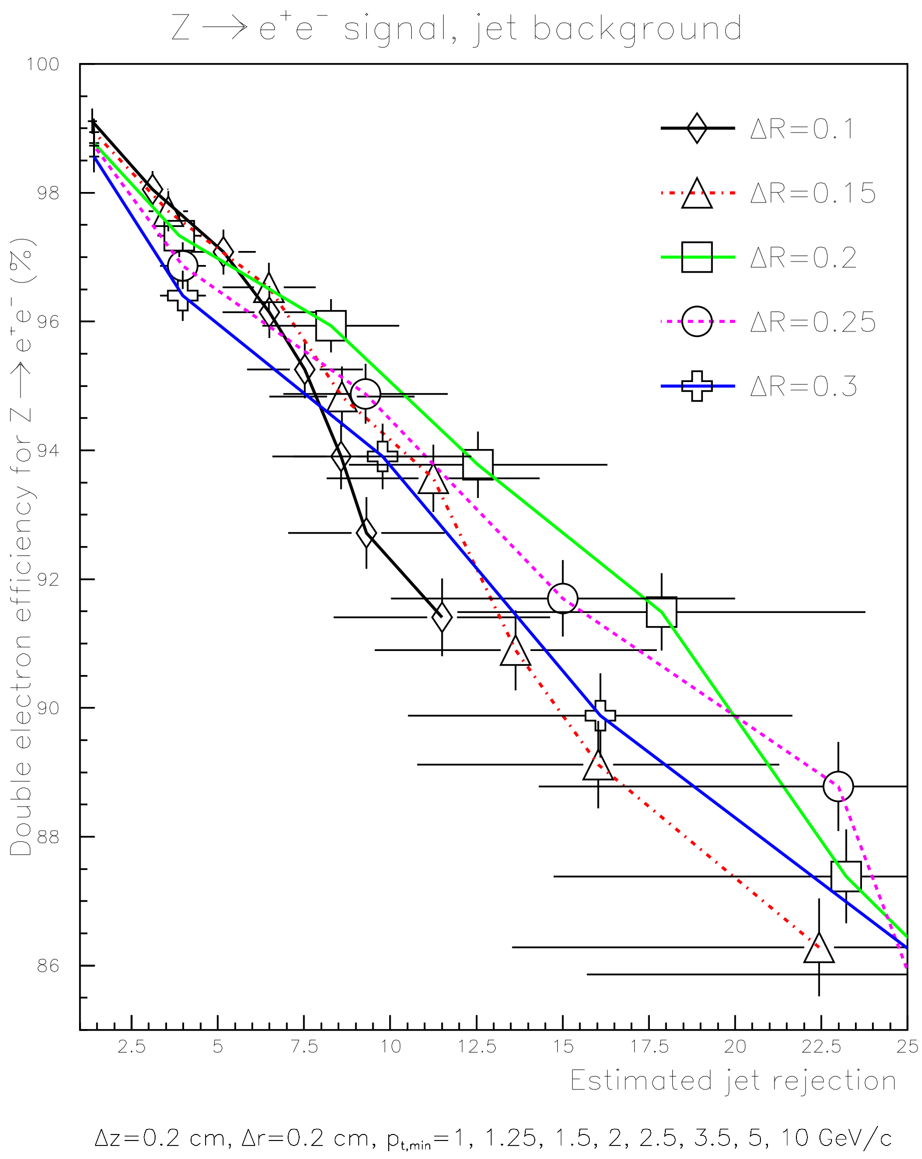


Figure 4.6: *Double-electron efficiency versus estimated background rejection for $Z \to e^+e^-$ signal compared to weighted jet background bins. In this plot, only signal events containing exactly two electron candidates have been considered. The background is estimated from single-electron efficiency with $P_{2e} = P_e^2$, using bin weights obtained from double-electron jets. The black error bars indicate 1σ statistical error.*

where the timer was called.

The number of loose Level-1 events that pass Level-2.5 selection and also have an electron candidate is quite small, only 187 events (157 with single track) for the almost full available loose Level-1 sample (27 924/29 194 events). To improve statistics, samples from the most important p_t bins (3050, 50100 and 100up) were timed without the additional *.orcarc* cuts mentioned in Table 4.2. Using looser de-

Table 4.4: *CPU consumption per track/event in background single-electron candidate events for a 450 MHz Pentium III [46]. The sample was 50100 jet bin with Level-1, Level-2 and Level-2.5 selection cuts made in ORCA with loose default settings. The isolation parameters were $p_t = 2.0$ GeV/c, $dR = 0.25$, $\Delta z = 0.2$ cm and $\Delta r = 0.2$ cm.*

Jet bin	Mean (s)	RMS (s)	nb of events	Sample	Level-2.5 weight
3050	0.96	0.78	165	2000	29%
50100	1.2	1.1	377	4000	60%
50100cut	1.2	0.94	8	2000	-
100up	2.1	1.9	97	733	11%
total	1.2	1.1	-	-	100%

fault settings increases the number of events passing Level-2.5 more than tenfold. It was assumed that the extra events will have roughly the same distribution of processing times as the events passing tighter cuts but the improvement in statistical error will be significant. This was also tested and found to hold for a smaller subsample.

Table 4.4 shows average processing times and RMS values for the different bins and for a smaller test sample of the 50100 bin. The test sample seems to validate the assumption that processing times are not significantly changed by the cuts. The overall processing time is calculated as a Level-2.5 weighted average of the processing times for different bins. The CPU time distribution for the 50100 bin is shown in Figure 4.7.

The main contribution to the processing time is expected to arise from the combinatorial track finding when triples of silicon pixel hits inside the isolation region are fitted to tracks with given constraints of minimum p_t, vertex position and direction of momentum. The computing time should increase linearly with the number of possible hit combinations and thus roughly exponentially with the available hits inside the isolation region. If the number of tracks and thus the number of hits inside the region is taken to be Gaussian distributed, we should obtain a log-Gaussian distribution of computing times, as is observed in the lower plot of Figure 4.7. This plot shows a Gaussian fit to the $\log_{10}$ of processing time. The statistical parameters for a Maximum Likelihood fit of the $\log_{10}$-Gaussian are: constant$=21.7 \pm 1.4$, $\mu = -0.058 \pm 0.018$, $\sigma = 0.347 \pm 0.013$ and the measure of the significance of the fit, $\chi^2 = 1.28$ (ideally 1).

Due to the exponential dependence of the processing time on the number of hits, the algorithm can be very sensitive to the size of the isolation region. The sample distribution in Figure 4.7 was obtained for a well representative, although not optimal, set of parameters. The average processing time is close to 1 second per electron candidate for a medium performance PC, which is within reasonable limits. The CPU time contribution from EgammaTrackIsolation is diluted by the mass of events failing Level-2.5 and the eventual contribution per Level-2 event is on the order of 100 milliseconds.

Significantly increasing the size of the isolation region is not expected to be necessary to improve the rejection performance, although small upward changes should still be in tolerable limits with respect to the processing time. On the other hand, the vertex search area $\Delta z \times \Delta r$ can be decreased by a factor of 8 without loosing more than 1–1.5 points in the jet rejection at 95% efficiency and at essentially no loss at efficiencies over 97–98%. This was tested to yield an equivalent reduction of roughly a factor of 2 in CPU time for the 50100 bin. Corresponding increase of the conesize times the vertex search area $\Delta R \times \Delta z \times \Delta r$ by a factor of 10 increases processing time by roughly a factor of 4 for the 50100 bin.

4.3.4 Performance compared to other available schemes

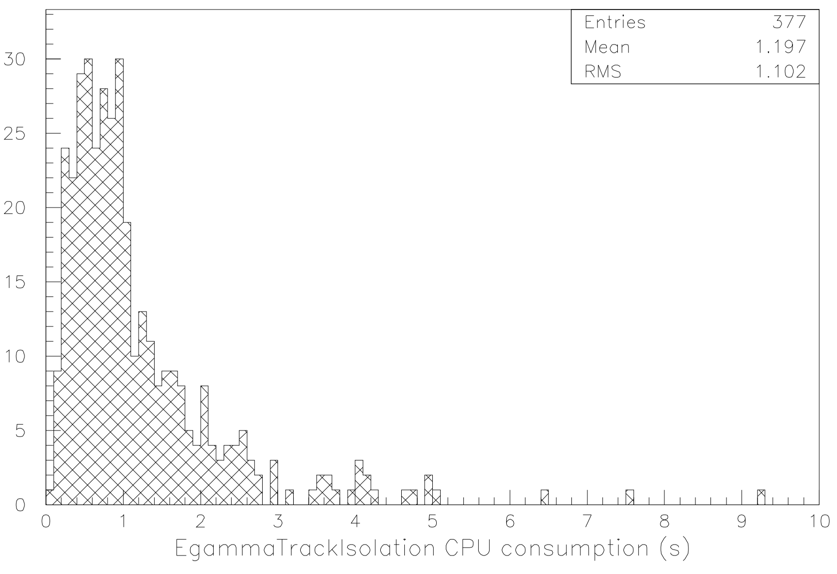
The CMS software is due at 2007 when the LHC experiments will xstart, four years from now. The LHC is planned to start at $2 \cdot 10^{33}$ cm^{-2}s^{-1} low luminosity drive and to reach its nominal high luminosity of 10^{34} cm^{-2}s^{-1} after year or two of successful operation. The current ORCA software development has thus focused mainly on completing the software on those parts required for the Data Acquisition and High-Level Trigger Technical Design Report (DAQ TDR) [23] and the initial low luminosity drive.

Level-2 ECAL isolation is expected to be used already at the low-luminosity run. The track/pixel-line isolation tools, on the other hand, are expected to be used mainly for the high-luminosity runs when the event rates are much higher and extra cuts on the data rate are needed. Simultaneously, the performance of ECAL isolation at Level-1 and also at Level-2, if implemented, is reduced due to the increased pile-up and cell occupancy. The same performance loss applies for Level-2.5 pixel matching, see Figure 4.8. Electron isolation selection cuts based on pixel lines or full track reconstruction is, however, expected to be much less affected by the pile-up.

At the moment, different approaches for implementing the electron isolation selection cuts at high luminosity are being considered, the algorithm introduced in this Thesis being one among them. Therefore, a comparison of the performance to competing/complementary solutions is an essential part of this work.

The main alternative solution to full track reconstruction is the pixel line isolation. Other possible schemes to cut on the data rate would be to use refined ECAL isolation at Level-2 [48] (Figure 4.9) and to fine-tune the already implemented Level-2.5 pixel matching [47] (Figure 4.8) to obtain higher jet rejection. Unfortunately, these alone seem to lead to unacceptably high loss in electron efficiency due to the increased cell occupancy at ECAL and pile-up at the pixel detector. It is therefore considered more rational to use either pixel line or track isolation or a similar technique that is only little affected by pile-up, at the expense of more CPU resources.

The pixel line isolation would be implemented either after Level-2.5 pixel matching or after Level-3, identically to the track isolation. It should be noted that the results presented here are not final and are subject to change since the final Level-1, Level-2 and Level-2.5 thresholds have not yet been fully agreed on. However,



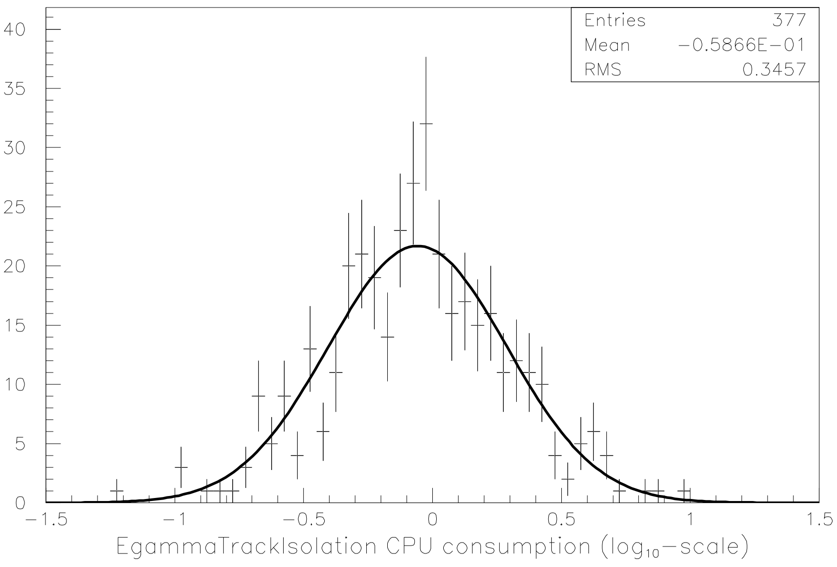


Figure 4.7: *EgammaTrackIsolation CPU consumption on a 450 MHz Pentium III with parameters $\Delta R = 0.25$, $\Delta r = 0.2$ cm, $\Delta z = 0.2$ cm, $p_{t,\min} = 2.0$ GeV/c. The sample was 50100 jet bin with Level-1, Level-2 and Level-2.5 selection cuts made in ORCA with loose default settings.*

the best estimates for the final values have been used, as shown in Table 4.2, and the results should provide a useful insight of the relative performances.

Figure 4.9 shows the performance for pixel line isolation as presented in DAQ TDR [23]. The signal and background samples, luminosity and placement in the selection cut chain are the same as those used for full track isolation in Figure 4.4. The x-axes used in plots are different and so are the scales, so a few representative values have been tabulated in Table 4.5.

In pixel line isolation the minimum p_t was fixed to 1 GeV/c but varied from

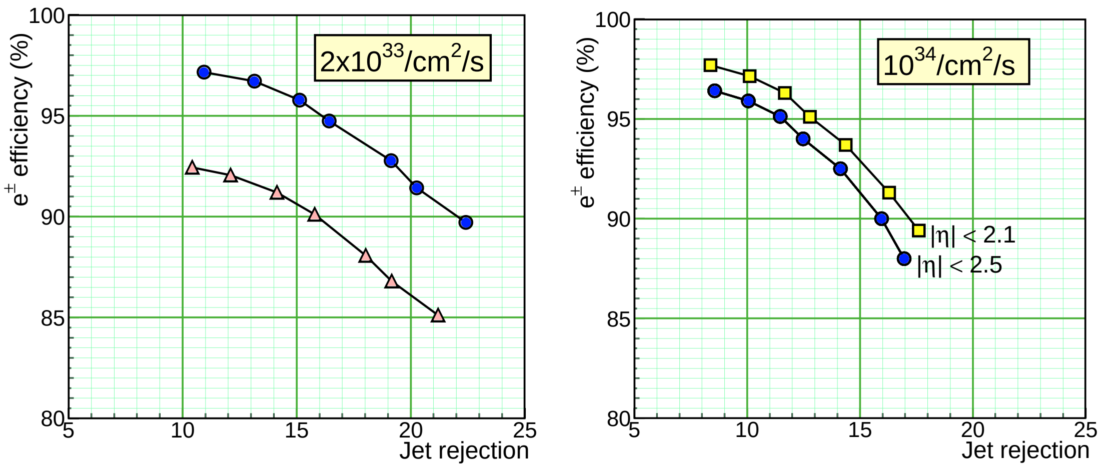


Figure 4.8: *Rejection versus efficiency obtained from Level-2.5 pixel matching. Left: at low luminosity (2×10^{33} cm^{-2}s^{-1}); the top curve shows the performance when the full pixel detector is used while the lower curve shows the performance for the staged scenario (the outer barrel layer and the outermost pixel endcap disks will not be present at start-up). Right: At high luminosity (10^{34} cm^{-2}s^{-1}) for two different pseudorapidity ranges. [47]*

1 to 10 GeV/c in track isolation. This is taken into account in the comparison. Also, the size of the vertex search area $\Delta z \times \Delta r$ was not quoted in the DAQ TDR. Values $\Delta z = 0.2$ cm and $\Delta r = 0.2$ cm were chosen for the electron track isolation. These choices would seem to be quite close to the ones used for pixel isolation since the electron efficiencies for $p_{t,\min} = 1$ GeV/c case are almost identical, as can be seen in Table 4.5. However, there is a quite large difference, 16–20%-unit in jet efficiency in favour of track isolation. Although a few percent of this is likely to be attributed to the better p_t resolution of full track reconstruction there is another likely scenario to account for most of the difference, explained below.

The pixel isolation algorithm uses only 3-pixel lines, i.e., the hits are required to be found on all the three pixel layers. If the pixel detectors are assumed to have an efficiency of 95% for recording a hit each, for three hits in three layers this results in an 86% efficiency. This will not, however, show up in the signal efficiency because only events where the pixel line or “pixel track” could be reconstructed are considered. The same argument goes, of course, for the jet-electron track that triggered the event. For all the other tracks inside the isolation cone there is a 14% possibility of losing them just because one of the hits was not recorded.

If only two hits out of three were required, the possibility of finding a track would be $3 \times 0.95 \times 0.95 \times 0.05 \times 100\% \approx 13.5\%$-units higher. The electron efficiency would be expected to stay the same because it is normalised to events having a track, but the jet rejection should increase by an amount corresponding to about a 13.5%-unit decrease in the jet efficiency. This is strictly valid only if all the signal events have zero and jet events exactly one extra track inside the isolation cone. Some corrections will also step in if more than one track per cone are considered.

Table 4.5: *Pixel line isolation versus track isolation. The values $p_t = 1$ GeV/c and $n = 1$ are assumed for both, unless otherwise stated. For the track-isolation vertex search area, the values $\Delta z = 0.2$ cm, $\Delta r = 0.2$ cm were used. The values in parenthesis are the efficiencies corresponding to the rejection factor.*

R_{cone}	Pixel efficiency	Pixel rejection	Track efficiency	Track rejection
0.2	91%	2.8 (0.36)	90.8±0.6%	6.3±1.5 (0.16)
0.3	82%	3.7 (0.27)	82.2±0.9%	9.5±2.6 (0.11)
0.3*	96%	2.2 (0.45)	96.4±0.4%	3.4±0.6 (0.29)
0.4	71.5%	4.3 (0.23)	72.4±1.1%	13.3±4.5 (0.075)

$^*n = 2$

Nevertheless, this should be the main reason why track isolation performs better than pixel line isolation. The NewSeedGenerator class used in track isolation uses pairs of pixel hits for creating track seeds and thus has essentially the same functionality as pixel-line reconstruction using only two hits.

If the 2-hit approach could be successfully adopted to pixel-line isolation, the two algorithms would have a closely comparable performance. The track isolation would still have slightly better p_t and position resolution which might improve the performance somewhat, but this would be unlikely to justify the higher CPU consumption. However, it is possible, even quite likely, that ghost tracks caused by random pile-up hits will become a major problem for pixel isolation and degrade the electron efficiency, degrading the increased jet rejection power, if the 2-hit approach were to be used. In full track reconstruction, the ghost seeds corresponding to pixel isolation ghost tracks are naturally eliminated when the track is propagated to the silicon strip layers.

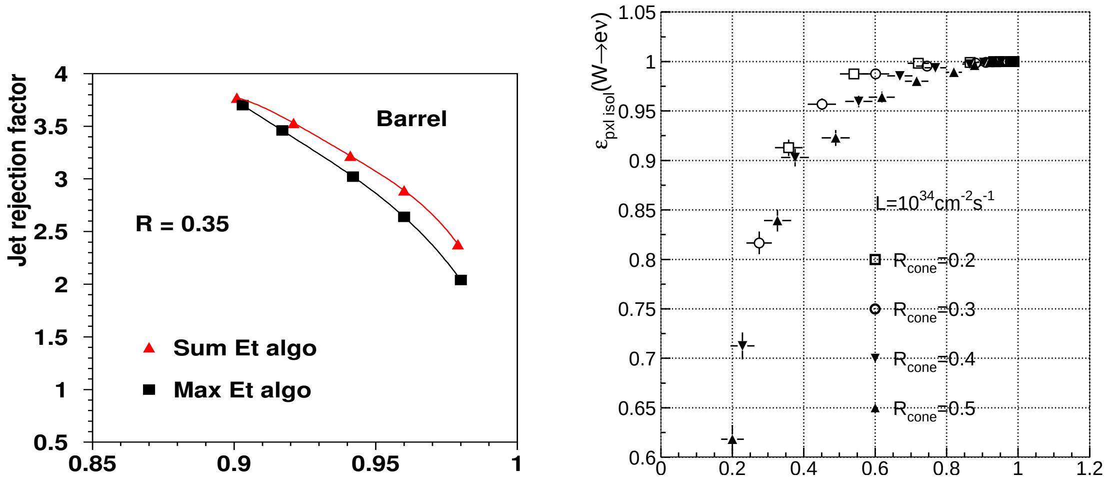


Figure 4.9: *Left: ECAL isolation performance using different algorithms [48]. Right: Rejection against jet background versus the efficiency for electrons from Ws when a pixel line isolation cut is applied after Level-2.5 selection at 10^{34} cm^{-2}s^{-1} [23].*

For the pixel-line isolation, only tracks with $p_t > 1$ GeV/c have been used. Tuning is done by varying the required number of tracks inside the isolation cone and the cone size but not the minimum p_t. This is a sensible approach since the p_t resolution using pixel lines only is quite poor. On the other hand, reaching high rejection factors with a high number of tracks is difficult and requires a relatively large cone size. This may easily lead to a large number of hit combinations and waste CPU, even if a single computation is cheap. Also, it seems that a single extra track yields the best efficiency versus rejection performance. It is for these reasons that the required track number is fixed to two (one extra track) in track isolation. The p_t resolution for full track reconstruction is much improved; therefore, a minimum p_t cut is used, instead for tuning. For pixel-line isolation, the cone size is not as much an issue since all possible pixel lines have already been constructed when PixelReconstruction is automatically called for each event, starting as of ORCA 6.3.0. Iteration over all pixel lines and simple testing of whether or not they fit inside the cone is then essentially a "no-cost" computation for any reasonable number of pixel lines.

The comparison of the CPU consumption of track isolation and pixel-line isolation is not as straightforward as it may first seem. The timing of track isolation is easy, simply turning the clock on when the isolation code is called and turning it off when the function returns. For pixel-line isolation this same approach generally gives a few nanoseconds, compared to seconds in the track isolation. The reason for this was already described above: iterating the already constructed pixel lines consumes essentially no time. The real hidden cost is the call to PixelReconstruction whenever a new event is analysed. This automatic feature was included in ORCA 6.3.0 but was not yet present in ORCA 6.2.3. However, in ORCA 6.3.0 the CPU time of PixelReconstruction is also automatically histogrammed, such that comparisons can be made easily.

Table 4.6 shows the CPU time needed to make the pixel reconstruction. The data was gathered on the same runs as the CPU measurements of track isolation in Table 4.4 were made, hence the two are directly comparable. Considering the raw times, regional track reconstruction is not more than a factor of two slower than complete pixel-line reconstruction. We have also to bear in mind that the regional track reconstruction considers also 2-hit seeds, whereas the pixel-line reconstruction is currently limited to 3-hit lines. The extra combinatorics involved in 2-hit lines would certainly also slow down pixel-line reconstruction. On the other hand, if more than one track per event is tested for isolation, pixel line isolation can be run with essentially no extra cost, in contrast to track isolation.

In conclusion, with the current implementations of the algorithms the track isolation is roughly a factor of two slower in terms of the CPU usage, but offers a significantly higher jet rejection (corresponding to 15–20%-unit decrease in jet efficiency) for the same signal efficiency. The CPU cost needed to run track isolation is around 1 second per track on a 450 MHz Pentium III [46], or around 100 milliseconds when normalised to Level-2. It can be reasonably expected that the signal rejection of the two algorithms overlaps considerably, thus a possible CPU saving approach could be to combine the two.

Table 4.6: *CPU consumption of PixelReconstruction per event in single-electron events on a 450 MHz Pentium III [46]. The sample was 50100 jet bin with Level-1, Level-2 and Level-2.5 selection cuts made in ORCA with loose default settings. The isolation parameters were $p_t = 2.0$ GeV/c, $dR = 0.25$, $\Delta z = 0.2$ cm and $\Delta r = 0.2$ cm.*

Jet bin	Mean	RMS	nb of events	Sample	Level-2.5 weight
3050	0.57	0.24	165	2000	29%
50100	0.52	0.23	377	4000	60%
50100cut	0.61	0.36	8	2000	-
100up	0.51	0.21	97	733	11%
total	0.53	0.23	-	-	100%

4.4 Extension to photon isolation

Although a photon-isolation algorithm is not one of the direct goals of this Thesis, some discussion on it is included because of the relative ease with which the electron isolation algorithm can be extended to the photon case and the importance of an implementation of a photon-isolation algorithm. After all, $H \to \gamma\gamma$ is expected to be one of the most promising channels for finding a light standard model Higgs boson.

Let us first state some differences between the electron candidates and photon candidates. By our current definition, electron candidates are $RecTrack$s obtained from RecEvent $EPTracks$ that contain an ECAL super-cluster and pixel detector hits matched to it. They do not contain a fully reconstructed primary vertex but can be interrogated for the impactPointState z-coordinate, which is as close to the vertex coordinates as you need to get, and also for the direction of the track at (or very near) the vertex. These are all that is needed to run the regional track reconstruction in EgammaTrackIsolation.

The photon candidates can in this case be defined as ECAL $EgammaSuperCluster$s in events that pass Level-2 but fail at Level-2.5. They also have to pass higher E_T thresholds than at Level-2. Here we have used a symmetric 25 GeV threshold for the double-photon channel. The definition may be later refined to mean $PhotonCandidate$ objects that have similar properties. Since unconverted photons do not leave any tracks to the pixel or strip detectors, the photon candidates cannot be assigned any vertex or impactPointState. To get around this problem, we can use the $PixelReconstruction$ class to find all possible vertices in the event.

```
PixelReconstruction
   pixelReconstruction(doVertexConstraint=true, doVertexFit=true,
                       doPtFit=true, doOverlapFlag=true,
                       do2HitRecovery=false, doRecHitCleaning=false,
                       THREE_HIT_RECO);
int status = pixelReconstruction.doIt(PT_MIN=1.0,MAX_Z_OFFSET=0.15);
const map<int, PixelVertex, less<int> >&
  vertex = pixelReconstruction.getVertices();
```

Whether or not we get the photons vertex right is not crucial; it should not contain any tracks to the direction of the super-cluster, anyhow. The efficiency will be only slightly degraded by those extra vertices that by chance have a track or two in the isolation region. What matters, is that we have the vertex of the fake photon among our vertex candidates. This vertex, in contrast to the photon vertex, should have some tracks pointing towards the super-cluster.

When we have a vertex candidate, we have solved half of the problem. The other half is an easy exercise in geometry to calculate the direction from the vertex to the super-cluster. Vector arithmetic gives

```
HepPoint3D scp = sc->Position(); // EgammaSuperCluster *sc
GlobalPoint vtx(0,0,z); // z position of the vertex
GlobalVector mom(scp.x()-vtx.x(),scp.y()-vtx.y(),scp.z()-vtx.z());
```

This slightly cumbersome notation contains the simple statement $\vec{p} \upuparrows \Delta\vec{x} = \vec{x}_{sc} - \vec{z}_{vtx}$, for $\vec{p}$ momentum, $\vec{x}_{sc}$ super-cluster position and $\vec{z}_{vtx}$ vertex projection on the z-axis. Regional track reconstruction can now be used in exact analogy to the electron case to obtain photon isolation for the vertex candidate. The overall isolation is obtained by iterating over all reconstructed vertices.

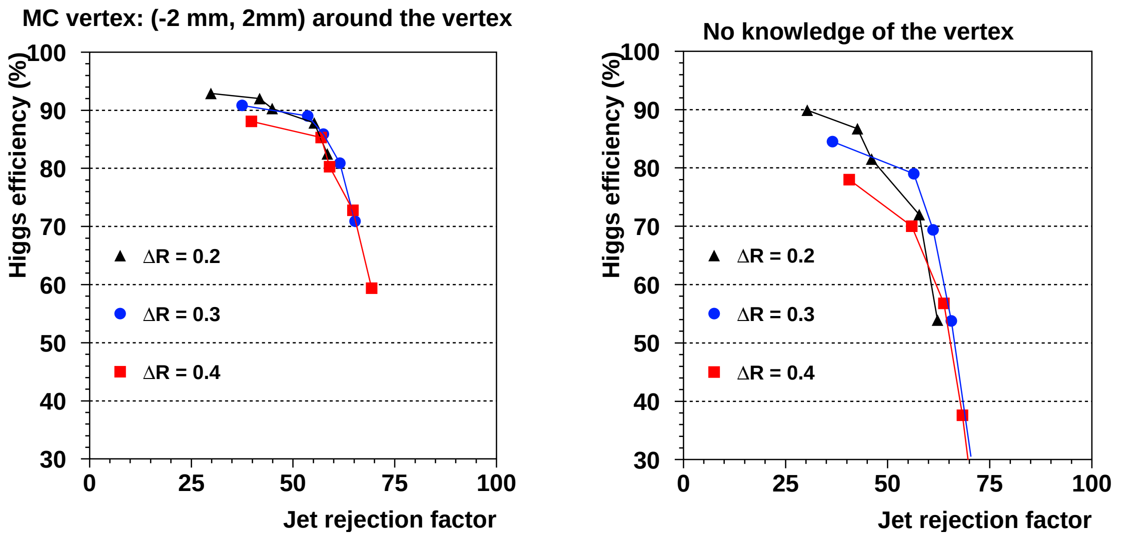


Figure 4.10: *Double-photon efficiency versus jet rejection for $H \rightarrow \gamma\gamma$ signal and QCD jet background, as reported in DAQ TDR [23]. First approach (left) uses the Monte Carlo vertex, second (right) all the reconstructed vertices.*

One set of results for the rejection of jet background in the double photon stream is shown in Figure 4.11. The double photon channel was chosen for study instead of the single photon case because it is exactly this channel that is important for finding the Higgs boson. The statistics are slightly worse than for the single photon channel but many other uncertainties are avoided. Compared to extrapolating double-photon performance from single-photon performance, different rejection factors for the first and the second photon are accommodated naturally and the correct bin weights follow directly from the calculations. The E_T thresholds for the super-clusters were symmetric, 25 GeV and 25 GeV, instead of the asymmetric $E_T^1 = 35$ GeV and $E_T^2 = 20$ GeV.

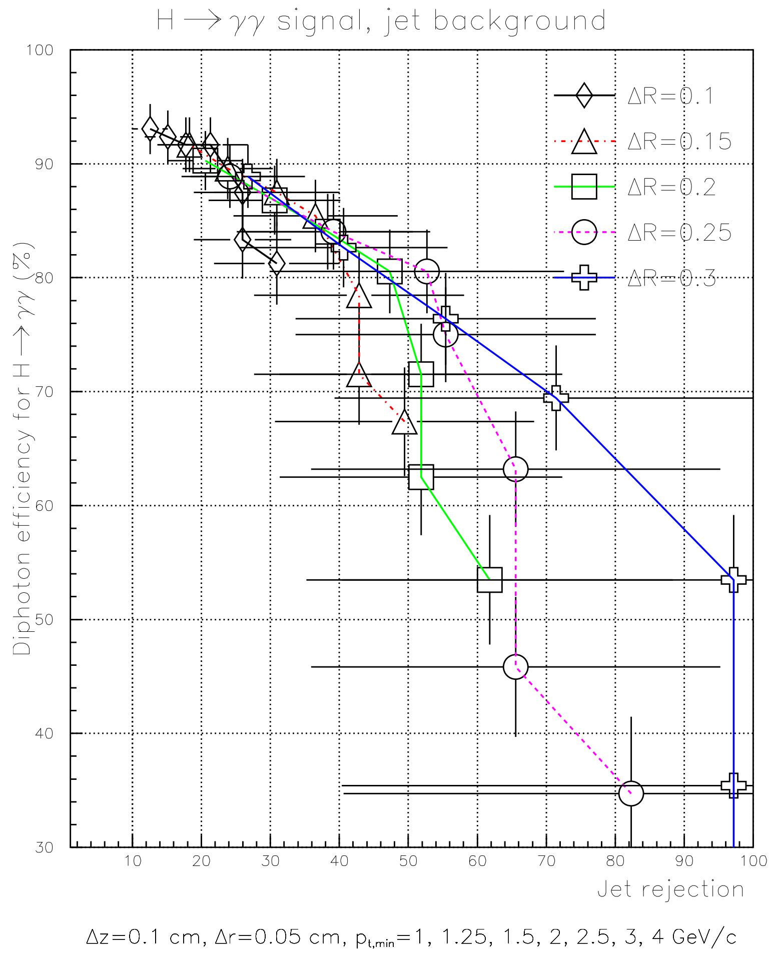


Figure 4.11: *Double-photon efficiency versus jet rejection for the $H \rightarrow \gamma\gamma$ signal and a QCD jet background. The results were obtained using EgammaTrackIsolation and vertices reconstructed with the PixelReconstruction class. The set of parameters was chosen to give results comparable to Figure 4.10 and is not optimal.*

The results in Figure 4.11 show diphoton performance comparable to the one reported in DAQ TDR, see Figure 4.10. The efficiency versus jet rejection remains quite similar in the high diphoton efficiency range, $\geq 80\%$. The power of track reconstruction is revealed at lower efficiencies and higher jet-rejection factors, as the rejection factors continue to climb, in contrast to the pixel isolation. Unfortunately, the statistics becomes a limiting factor and the jet-rejection factors cannot be accurately determined above 40–50, as indicated by the long error bars.

The CPU consumption per “photon” track, i.e. per line from vertex candidate to super-cluster, for the EgammaTrackIsolation is on the same order as per electron track, slightly less if the parameters are also optimised for CPU consumption. The total number of vertices reconstructed per event is usually 10–25, depending on the vertex-finding parameters. Multiplied by two super-clusters, this would have a price in terms of the CPU time usage, unless 1–3 vertices are preselected successfully. For easy usage with the external vertex finders, EgammaTrackIsolation includes the user interface

```
EgammaTrackIsolation->getTrackCount(EgammaSupercluster *sc,
                                    GlobalPoint zvtx, int ptr);
```

that allows the user to input the vertex coordinate, bypassing the vertex-finder algorithm.

Chapter 5

Summary and discussion

In this Thesis, an algorithm for electron isolation in the CMS data analysis was designed, implemented and tested. The program was submitted to the CMS CVS repository and default values were set for the parameters used in the algorithm. The EgammaTrackIsolation class is to be included in the Level-3 selection code, L3Selection, that will be released soon.

The track reconstruction was shown to be an efficient method for electron isolation, with a typical electron efficiency of 96.3±0.4% for a $W \rightarrow e\nu$ sample versus QCD jet background rejection performance of 4.0±0.8 (default parameters). A slightly different configuration was tested to consume on average 1.2 s of CPU time with an RMS of 1.1 s per electron track on a 450 MHz Pentium III. The CPU time distribution was shown to have a log-Gaussian distribution, confirming the assumption that the computation times depend exponentially on the number of hits inside the isolation cone, and thus also on the size of the isolation region $\Delta R \times \Delta z \times \Delta r$.

The main competing/complementary method for testing electron isolation is pixel line isolation. Its performance in Figure 4.9, in a case similar to track isolation, is 96% efficiency for 2.2 jet rejection. The CPU consumption for pixel line isolation was tested to depend only on the time it takes to do a full pixel reconstruction for the event. This consumes on average 0.53 s CPU time with an RMS of 0.23 s per event for the default configuration.

The performance for a $Z \rightarrow e^+e^-$ sample was also tested, but several approximations and estimates had to be introduced in order to overcome the low statistics of the double-electron background. The results for double-electron events were 91.5±0.6% event efficiency versus 17.9±5.9 background rejection with default parameters. It should be stressed that these results are only rough estimates, as the single-electron background was used to estimate double-electron rejection.

An extension of the electron isolation algorithm was tested for photon isolation. The main channel of interest here is $H \rightarrow \gamma\gamma$, so only double-photon samples were used. A typical result from Figure 4.11 is 85.4±3.2% event efficiency for $H \rightarrow \gamma\gamma$ versus 37±12 rejection of jet background in the double-photon stream. The CPU consumption per track is of the same order as for a single electron, and in a typical event there are 10–25 vertices and two super-clusters, giving 20–50 times more CPU consumed per event for double photons than for single electrons. For this reason the default values were set for a more moderate performance, giving 91.7±2.4% event efficiency at 21.3±5.4 jet rejection and 5–10 s CPU consumption per event. It is also possible to bypass the pixel reconstruction and vertex iteration and input the vertex directly.

In conclusion, it was shown that track isolation is a well-performing method for determining electron isolation, to be used either alone or as a complementary method with pixel line isolation. Future developments of the algorithm will most likely concentrate on maintaining compatibility with new versions of ORCA, as the program is based on a high level of abstraction and classes are constantly updated. A new ORCA 7 compatible version has already been committed, but is not covered in this Thesis.

The new results in this Thesis will help towards optimising the extraction of interesting data from the 800 million inelastic collisions per second detected in the CMS. Only about 100 events per second can be recorded on disk and fully analysed, all the rest of the events will be irreversibly lost. This huge reduction rate thus means that selection algorithms will play a key role when new physics data is collected.

Be it the Higgs boson, supersymmetry or something completely unexpected, theorists are convinced that interesting physics will be found in the pristine TeV range first explored by the LHC. The new discoveries ahead should have an impact on the theoretical front-line of particle physics, and they may also impact cosmology that searches an explanation for dark matter and dark energy.

Bibliography

[1] On CERN's name, see: http://user.web.cern.ch/user/cern/CERNName.html.

[2] The Compact Muon Solenoid Technical Proposal, CERN/LHCC 94-98, CERN/LHCC/P1, 15 December, 1994.

[3] ATLAS Technical Proposal, for a General-Purpose pp Experiment at the Large Hadron Collider at CERN, CERN/LHCC/94-43, CERN/LHCC/P2, 15 December, 1994.

[4] LHCb Technical Proposal, A Large Hadron Collider Beauty experiment for precision measurements of CP-violation and rare decays, CERN/LHCC/98-4, CERN/LHCC/P4, 20 February, 1998.

[5] ALICE Technical Proposal, for A Large Ion Collider Experiment at the CERN LHC, CERN/LHCC/95-71, LHCC/P3, December, 1995.

[6] S. Glashow, Nucl. Phys. **22**, 579 (1961).
S. Weinberg, Phys. Rev. Lett. **19**, 1264 (1967).
A. Salam in *Elementary Particle Theory*, ed. by N. Svartholm, Aspenäsgården, 367 (1968).

[7] F. Halzen, A. D. Martin, Quarks & Leptons: An Introductory Course in Modern Particle Physics, John Wiley & Sons, 1984.

[8] E. B. Karlsson in *The Nobel Prize: The First 100 Years*, ed. by A. W. Levinovitz and N. Ringertz, Imperial College Press and World Scientific Publishing, 2001.

[9] R. Kleiss, Fundamental Concepts of Particle Physics (16), CERN Summer Student Lectures, 2002.

[10] K. Hagiwara *et al.*, Physical Review **D 66**, 010001, 2002.

[11] P. Higgs: Phys. Lett. **12**, 132 (1964); Phys. Rev. Lett. **113**, 508 (1964); Phys. Rev. **1145**, 145 (1966).
F. Englert and R. Brout, Phys. Rev. Lett. **13**, 321 (1964).
G. S. Guralnik, C. R. Hagen and T. W. Kibble, Phys. Rev. Lett. **13**, 585, (1964).

[12] G. Wrochna, Challenges of CMS, a popular talk, CMS Outreach: http://cmsdoc.cern.ch/cms/outreach/html/CMSdocuments/CMSdocuments.html.

[13] Higgs Physics in CMS, CMS Outreach: http://cmsdoc.cern.ch/cms/outreach/html/CMSdetectorInfo/HiggsPhysics/

[14] W. E. Burcham, M. Jobes, Nuclear and Particle Physics, Longman Group Limited, 1995.

[15] ALEPH, DELPHI, L3 and OPAL Collaborations, The LEP Working Group for Higgs Boson Searches, Search for the Standard Model Higgs Boson at the LEP, CERN-EP/2003-011, 2003.

[16] The LEP Electroweak Working Group, http://lepewwg.web.cern.ch/LEPEWWG/

[17] P. Fayet and S. Ferrara, Phys. Rep. **32**, 249 (1977).
H. P. Nilles: Phys. Rep. **110**,1 (1984).
H. Haber and G. Kane: Phys. Rep. **117**, 75 (1985).
S. Martin: hep-ph/9709356 (1999).

[18] H. P. Nilles, Phys. Rep. **110**, 1 (1984)
S. Dawson: hep-ph/9712464 (1997).

[19] The Large Hadron Collider Conceptual Design, CERN/AC/95-05 (LHC), 20 October, 1995.

[20] CMS Transparencies, CMS Outreach: http://cmsdoc.cern.ch/cms/outreach/html/CMSdocuments/CMSdocuments.html.

[21] Tracker Technical Design Report, CERN/LHCC 98-6, CMS TDR 5, 15 April, 1998.

[22] Addendum to Tracker Technical Design Report, CERN/LHCC 2000-016, CMS TDR 5 Addendum 1, 21 February, 2000.

[23] The TriDAS Project Technical Design Report, Volume 2: Data acquisition and High-Level Trigger, CERN/LHCC 2002-26, CMS TDR 6.2, 15 December, 2002.

[24] The Electromagnetic Calorimeter Project Technical Design Report, CERN/LHCC 97-33, CMS TDR 4, 15 December, 1997.

[25] The TriDAS Project Technical Design Report, Volume 1: The Trigger Systems, CERN/LHCC 2000-038, CMS TDR 6.1, 15 December, 2000.

[26] Technical Proposal for CMS Computing, CERN/LHCC 96-45, 19 December, 1996.

[27] Interactive Graphical User ANAlysis (IGUANA),
http://iguana.cern.ch.

[28] High-energy-physics event generator Pythia,
http://www.thep.lu.se/∼torbjorn/Pythia.html.

[29] CMS Simulation and Reconstruction Package (CMSIM),
http://cmsdoc.cern.ch/cmsim/cmsim.html.

[30] GEANT (Geometry And Tracking) detector description and simulation tool (version 3),
http://wwwinfo.cern.ch/asd/geant.

[31] Object-oriented Reconstruction for CMS Analysis (ORCA),
http://cmsdoc.cern.ch/orca.

[32] Physics Analysis Workstation (PAW),
http://wwwinfo.cern.ch/asd/paw.
O. Couet, PAW Tutorial, CERN, 1999.
Information Technology Division, PAW User's Guide, CERN Program Library Long Writeup Q121, 1992-1999.

[33] Object oriented Simulation for CMS Analysis and Reconstruction (OSCAR),
http://cmsdoc.cern.ch/oscar.

[34] GEANT4 simulation tool,
http://wwwinfo.cern.ch/asd/geant4.

[35] CMS Fast Simulation (FAMOS),
http://cmsdoc.cern.ch/FAMOS.

[36] Coherent Object-oriented Base Reconstruction and Analysis (COBRA),
http://cobra.web.cern.ch/cobra.

[37] Detector Description Database (DDD),
http://cmsdoc.cern.ch/cms/software/ddd/www.

[38] Root - An Object-Oriented Data Analysis Framework,
http://root.cern.ch.

[39] Dependency structure of ORCA, automatically generated by doxygen for ORCA 6.2.0., 21 August, 2002.

[40] K. Lassila-Perini, Jet rejection with matching ECAL clusters to pixel hits, CMS NOTE 2001/021, May 22, 2001.

[41] Data From the 2002 production,
http://cmsdoc.cern.ch/Physics/egamma/www/CurrentData.html

[42] T. Sjöstrand, P. Eden, Ch. Friberg, L. Lönnblad, C. Miu, S. Mrenna and E. Norrbin, Computer Phys. Commun. **135**, 238 (2001).

[43] CMS Production page, http://cmsdoc.cern.ch/cms/production/www/html/general/indexSQL.html

[44] High luminosity cuts and thresholds at 10^{34} (8 kHz Level-1 scenario), http://cmsdoc.cern.ch/Physics/egamma/www/Cuts2002.html

[45] C. Seez and D. Barney, Summary of ECAL-egamma Autumn 2000 Monte-Carlo Data Production Parameters and Specifications, CMS Internal Note, CMS IN 2001/023, 4 July, 2001.

[46] The computer used for the CPU analysis was lxcmsb4.cern.ch, a 450 MHz Pentium III running RedHat 6.

[47] G. Daskalakis, K. Lassila-Perini, Jet Rejection Using the Pixel Matching for the Low and the High Luminosity, CMS NOTE-2002/039, 26 November, 2002.

[48] V. Litvin, H. Newman, S. Shevchenko, Isolation studies for electrons and photons based on ECAL, CMS Internal Note, CMS IN 2002/023, 26 April, 2002.

Appendix A

EgammaTrackIsolation source code

```
/* EgammaTrackIsolation.h - Rev 1.1
 */

#ifndef EgammaTrackIsolation_h
#define EgammaTrackIsolation_h

/** \Class EgammaTrackIsolation This class is used to
 *  get the number of tracks inside an isolation cone.
 *  The cone geometry is defined by ptMin, conesize,
 *  rspan and zspan. Found tracks (also invalid ones)
 *  can be histogrammed using IsoTrackInfo. Use
 *  the .orcarc card EGTI:HistoSwitch=1 to turn
 *  histogramming on (ITRK block).
 */

#include "TrackerReco/TkSeedGenerator/interface/SeedGeneratorFromPixel.h"
#include "TrackerReco/TkSeedGenerator/interface/NewSeedGenerator.h"
#include "TrackerReco/TkSeedGenerator/interface/RectangularEtaPhiTrackingRegion.h"

#include "TrackerReco/TkSeedGenerator/interface/TrackingRegionSimpleFactory.h"
#include "TrackerReco/GtfPattern/interface/CombinatorialTrackFinder.h"

#include "CommonReco/PatternTools/interface/RecTrack.h"
#include "ElectronPhoton/ClusterTools/interface/EgammaSuperCluster.h"

//class G3EventProxy;
class EgammaTrackIsolation
{

 public:

  /** First call creates a new instance.
```

```
   *  Following calls will return a handle
   *  to the instance.
   */
  static EgammaTrackIsolation* instance();

  virtual ~EgammaTrackIsolation();

  /// Get number of tracks inside an isolation cone using rt as seed.
  int getTrackCount(const RecTrack * const rt, int ptr=0);
  /// As above but use zvtx instead of rt's vertex.
  int getTrackCount(const RecTrack * const rt, GlobalPoint zvtx, int ptr=0);
  /// Get number of tracks inside an isolation cone using sc as seed.
  /// Iterates over all pixel vertices created by PixelReconstruction.
  int getTrackCount(const EgammaSuperCluster * const sc, int ptr=0);
  /// As above but use zvtx instead of iterating over all pixel vertices.
  int getTrackCount(const EgammaSuperCluster * const sc, GlobalPoint zvtx,
                    int ptr=0);


  /// Get pt cut for itracks.
  float getPtMin(bool getE=true) {
    if(getE) return ptMin;
    else return ptMinG; }
  /// Get isolation cone size.
  float getConeSize(bool getE=true) {
    if(getE) return conesize;
    else return conesizeG; }
  /// Get maximum ivertex z-coordinate spread.
  float getZspan(bool getE=true) {
    if(getE) return zspan;
    else return zspanG; }
  /// Get maximum transverse distance of ivertex from beam line.
  float getRspan(bool getE=true) {
    if(getE) return rspan;
    else return rspanG; }

private:

  // Default constructor
  EgammaTrackIsolation();
  // Call regional track reconstruction
  int regionalTrackReco(GlobalVector mom, GlobalPoint vtx, int ptr, bool isE);

  // Protected instance of the class itself
  static EgammaTrackIsolation *theinstance;
  // Classes used by regional track reconstruction
  CombinatorialTrackFinder *theTrackFinder;
  SeedGeneratorFromPixel<NewSeedGenerator> *theRegional;
  TrackingRegionSimpleFactory<RectangularEtaPhiTrackingRegion> *theRegionFactory;
  RectangularEtaPhiTrackingRegion *rectRegion;
  // other choices are: ConeTrackingRegion, GlobalTrackingRegion
  // Parameters of isolation cone geometry.
  // I Electron case
  float ptMin;
  float conesize;
  float zspan;
  float rspan;
  // II Photon case (G for Gamma)
  float ptMinG;
  float conesizeG;
  float zspanG;
  float rspanG;
  bool histo;

};
#endif
```

```
/* EgammaTrackIsolation.cc - Rev 1.1
 */

#include "Utilities/Configuration/interface/Architecture.h"

#include "ElectronPhoton/EgammaTracks/interface/EgammaTrackIsolation.h"
#include "ElectronPhoton/EgammaTracks/interface/IsoTrackInfo.h"

#include "ElectronPhoton/EgammaAnalysisFactory/interface/EgammaAnalysisFactory.h"
#include "Utilities/Notification/interface/Singleton.h"

#include "Utilities/UI/interface/SimpleConfigurable.h"
#include "CommonReco/PersistentTrack/interface/TTrack.h"
#include "Utilities/Notification/interface/TimingReport.h"
#include "TrackerReco/PixelReconstruction/interface/PixelReconstruction.h"
#include "TrackerReco/PixelReconstruction/interface/PixelVertex.h"


EgammaTrackIsolation* EgammaTrackIsolation::theinstance = 0;

EgammaTrackIsolation* EgammaTrackIsolation::instance()
{
  if(theinstance==0)
    theinstance = new EgammaTrackIsolation();
  return theinstance;
}

// Defaults good for (efficiency / jet rejection)
// electron: wenu vs. jet sample electrons  (96% / 4)
// (dbl) photon: hgg vs. jet sample diphotons (92%+-3 / 21+-6)
EgammaTrackIsolation::EgammaTrackIsolation() :
  ptMin(SimpleConfigurable<float>(2.0,"EgTrkIso:PtMin")),
  conesize(SimpleConfigurable<float>(0.2,"EgTrkIso:ConeSize")),
  zspan(SimpleConfigurable<float>(0.1,"EgTrkIso:ZSpan")),
  rspan(SimpleConfigurable<float>(0.2,"EgTrkIso:RSpan")),
  ptMinG(SimpleConfigurable<float>(2.0,"EgTrkIso:PtMinG")),
  conesizeG(SimpleConfigurable<float>(0.1,"EgTrkIso:ConeSizeG")),
  zspanG(SimpleConfigurable<float>(0.1,"EgTrkIso:ZSpanG")),
  rspanG(SimpleConfigurable<float>(0.05,"EgTrkIso:RSpanG")),
  histo(SimpleConfigurable<bool>(false,"EGTI:HistoSwitch"))
{
  theRegionFactory =
    new TrackingRegionSimpleFactory<RectangularEtaPhiTrackingRegion>;
  theRegional =
    new SeedGeneratorFromPixel<NewSeedGenerator>(theRegionFactory);
  theTrackFinder =
    new CombinatorialTrackFinder();

  cout << "EgammaTrackIsolation instance:"
       << " ptMin=" << ptMin << "|" << ptMinG
       << " conesize="<< conesize << "|" << conesizeG
       << " zspan=" << zspan << "|" << zspanG
       << " rspan=" << rspan << "|" << rspanG
       << endl;
}

EgammaTrackIsolation::~EgammaTrackIsolation()
{
  if (theRegionFactory) delete theRegionFactory;
  if (theRegional) delete theRegional;
  if (theTrackFinder) delete theTrackFinder;
}



int EgammaTrackIsolation::getTrackCount(const RecTrack * const rt, int ptr)
{
  TimeMe t("Reconstructing_EgammaTracks(e)");

  GlobalVector mom = rt->momentumAtVertex();
  GlobalPoint vtx(0,0,rt->impactPointState().globalPosition().z());

  int ntrak = regionalTrackReco(mom,vtx,ptr,true);
  return (ntrak-1);
}

int EgammaTrackIsolation::getTrackCount(const RecTrack * const rt,
                                        GlobalPoint zvtx, int ptr)
{
  TimeMe t("Reconstructing_EgammaTracks(e)");

  GlobalVector mom = rt->momentumAtVertex();
  // Just to insure consistency with no-vertex-code
  GlobalPoint vtx(0,0,zvtx.z());

  int ntrak = regionalTrackReco(mom,vtx,ptr,true);
  return (ntrak-1);
}


int EgammaTrackIsolation::getTrackCount(const EgammaSuperCluster * const sc,
                                        int ptr)
{
  TimeMe t("Reconstructing_EgammaTracks(p)");

  const bool doVertexFit = true;  // do a detailed PV fit
  const bool doPtFit = true;      // do the track pt fit
  const bool doVertexConstraint = true; // use  vertex constraint
  const bool doOverlapFlag = true; // delete ovelapping tracks
  const bool do2HitRecovery = false; // recover tracks with 2hits only
  const bool doRecHitCleaning = false;
  const float PT_MIN = ptMinG; // used to be 1.0 as default
  const float MAX_Z_OFFSET = 0.15; // perhaps change 0.15 to zspan
  PixelReconstruction pixelReconstruction(doVertexConstraint, doVertexFit,
                                          doPtFit, doOverlapFlag,
                                          do2HitRecovery, doRecHitCleaning,
                                          THREE_HIT_RECO);
  //  int status =
  pixelReconstruction.doIt(PT_MIN,MAX_Z_OFFSET);

  const map<int, PixelVertex, less<int> >&
    vertex = pixelReconstruction.getVertices(); // Get the vertex list

  int vtxSize = vertex.size();  // number of vertices
  cout <<" Number of vertices found = " << vtxSize << endl;
  map<int, PixelVertex, less<int> > :: const_iterator idVertex;

  // loop over vertices
  int ntrak=0;
  int ntraktemp=0;
  for (int i=0;i<vtxSize;i++) {

    idVertex = vertex.find(i);
    int i1 = (idVertex->second).getTracksUsed(); // num of tracks used in fit
    int i2 = (idVertex->second).getTracks();     // num of tracks assigned init
    float sumPt = (idVertex->second).getPt();         // summed pt
    float sumP = (idVertex->second).getPmom().mag();  // summed mom magnitude
    float z = (idVertex->second).getPosition().z();   // z position

    // vertex flag (PV=1)
    cout << " index, tracks, sum-pt, sum-p, z = "
         << (idVertex->second).getFlag() << " "
         << i1 << "/" // num of tracks used in the fit
         << i2 << " ";    // num of tracks assigned init
    cout << sumPt << " "         // summed pt
         << sumP << " "; // summed momentum magnitude
    cout << z << endl;     // z position

    // calculate direction to EgammaSuperCluster
    HepPoint3D scp = sc->Position();
    GlobalPoint vtx(0,0,z);
    GlobalVector mom(scp.x()-vtx.x(),scp.y()-vtx.y(),scp.z()-vtx.z());

    TimeMe tb("Recontructing_EgammaTracks(pN)");
    // Code pointer as 100*ncluster + nvertex
    ntraktemp = regionalTrackReco(mom,vtx,100*ptr+i,false);
    if (ntraktemp>ntrak) ntrak = ntraktemp;
  }
  return (ntrak);
}

int EgammaTrackIsolation::getTrackCount(const EgammaSuperCluster * const sc,
                                        GlobalPoint zvtx, int ptr)
{
  TimeMe t("Reconstructing_EgammaTracks(pvtx)");
  // calculate direction to EgammaSuperCluster
  HepPoint3D scp = sc->Position();
  // to insure consistency with no-free-vertex-code
  GlobalPoint vtx(0,0,zvtx.z());
  GlobalVector mom(scp.x()-vtx.x(),scp.y()-vtx.y(),scp.z()-vtx.z());

  // Code pointer to 100*cluster_indx + vertex_indx=0 as in gTC above
  int ntrak = regionalTrackReco(mom,vtx,100*ptr,false);
  return (ntrak);
}

int EgammaTrackIsolation::regionalTrackReco(GlobalVector mom, GlobalPoint vtx,
                                            int ptr, bool isE)
{
  TimeMe tc("Reconstructing_EgammaTracks");

  float EgConeSize = (isE) ? conesize : conesizeG;
  float EgPtMin    = (isE) ?    ptMin :    ptMinG;
  float EgRSpan    = (isE) ?    rspan :    rspanG;
  float EgZSpan    = (isE) ?    zspan :    zspanG;

  RectangularEtaPhiTrackingRegion::LeftRightMargin
    phiMargin(EgConeSize,EgConeSize); //
  RectangularEtaPhiTrackingRegion::LeftRightMargin
    etaMargin(EgConeSize,EgConeSize); //
  RectangularEtaPhiTrackingRegion
    rectRegion(mom, vtx, EgPtMin, EgRSpan, EgZSpan, etaMargin, phiMargin);

  theRegionFactory->setRegion(rectRegion);
  theTrackFinder->setSeedGenerator(theRegional);

  vector<RecTrack> isoTracks;
  theTrackFinder->reco(isoTracks);

  // Check that reconstructed tracks fit within cone boundaries,
  // i.e. "round the corners" of the rectangular region.
  // (Note: tracks will not always stay within boundaries
  //  even with a conical region)
  int ntrak = 0;
  for (vector<RecTrack>::const_iterator it=isoTracks.begin();
       it!=isoTracks.end(); it++)
    {
      GlobalVector imom = it->momentumAtVertex();
      GlobalPoint ivtx = it->impactPointState().globalPosition();
      float pt = imom.perp();
      float dperp = ivtx.perp()-vtx.perp();
      float dz = ivtx.z()-vtx.z();
      float deta = imom.eta()-mom.eta();
      float dphi = imom.phi()-mom.phi();
      // Correct dmom_phi's from [-2pi,2pi] to [-pi,pi]
      if (dphi>M_PI) dphi = dphi - 2*M_PI;
      else if (dphi<-M_PI) dphi = dphi + 2*M_PI;
      float R = sqrt( dphi*dphi + deta*deta );

      // Apply boundary cut
      if (pt > ptMin && R < conesize &&
          fabs(dperp) < rspan &&  fabs(dz) < zspan) ntrak++;

      // This histograms invalid tracks, too.
      if (histo) {
        // activate histogramming
        AbstractHistogrammer *h = 0;
        h = EgammaAnalysisFactory::instance()->getHistogrammer();

        IsoTrackInfo trackh(it,ptr,vtx.z());
        *h << trackh;
      }
    }

  return (ntrak);
}
```

```
/* IsoTrackInfo.h - Rev 1.1
 */

#ifndef IsoTrackInfo_H
#define IsoTrackInfo_H

/** \Class IsoTrackInfo This class is used for histogramming tracks
 *  found by EgammaTrackIsolation.
 *  Fills an ITRK block to the ntuple. Variables are as follows
 *  nitrk  - Array length [0,100]
 *  itchrg - itrack charge
 *  ittip  - itrack transverse impact parameter
 *  itlip  - itrack longitudinal impact parameter
 *  itptr  - pointer to parent track/supercluster index (user defined)
 *  itvh   - number of valid hits in itrack
 *  itih   - number of invalid hits in itrack
 *  izvtx  - z-vertex used for isolation cone (user defined)
 */
#include "ElectronPhoton/EgammaAnalysis/interface/EgammaHistogrammable.h"
class AbstractHistogrammer;
class RecTrack;

class IsoTrackInfo : public EgammaHistogrammable
{
public:
  /// Call this before starting histogramming to pass the handle *h.
  IsoTrackInfo(AbstractHistogrammer *h);
  /// Create a new histogram entry.
  IsoTrackInfo(const RecTrack * const tr, int ptr=-1000, float zvtx=-1000);
  /// Copy constructor.
  IsoTrackInfo(const IsoTrackInfo &b);

private:
  void init();
  void init(const IsoTrackInfo &b);
  void init(const RecTrack * const tr, int tptr, float tzvtx);

  // Track information
  Variable<int> charge;
  Variable<Hep3Vector> mom;
  Variable<float> chisq;
  Variable<float> tip;
  Variable<float> lip;
  Variable<int> ptr;
  Variable<int> vhits;
  Variable<int> ihits;
  Variable<float> zvtx;
};
#endif
```

```
/* IsoTrackInfo.cc - Rev 1.1
 */

#include "Utilities/Configuration/interface/Architecture.h"

#include "CommonReco/PersistentTrack/interface/TTrack.h"
#include "CommonDet/PatternPrimitives/interface/TrajectoryStateOnSurface.h"

#include "ElectronPhoton/EgammaTracks/interface/IsoTrackInfo.h"

IsoTrackInfo::IsoTrackInfo(AbstractHistogrammer *h) :
  EgammaHistogrammable("itrk",100,h)
{
  init();
}
IsoTrackInfo::IsoTrackInfo(const RecTrack * const tr, int ptr, float zvtx) :
  EgammaHistogrammable("itrk",100,0)
{
  init(tr,ptr,zvtx);
}
IsoTrackInfo::IsoTrackInfo(const IsoTrackInfo &b) :
  EgammaHistogrammable(b)
{
  init(b);
}

void IsoTrackInfo::init()
{
  charge.Set("itchrg",0,this);
  mom.Set("itp",Hep3Vector(),this);
  chisq.Set("itchi2",0.,this);
  tip.Set("ittip",0.,this);
  lip.Set("itlip",0.,this);
  ptr.Set("itptr",0,this);
  vhits.Set("itvh",0,this);
  ihits.Set("itih",0,this);
  zvtx.Set("izvtx",0,this);

}
void IsoTrackInfo::init(const IsoTrackInfo &b)
{
  init();
  charge = b.charge;
  mom = b.mom;
  chisq = b.chisq;
  tip = b.tip;
  lip = b.lip;
  ptr = b.ptr;
  vhits = b.vhits;
  ihits = b.ihits;
  zvtx =  b.zvtx;
}
```

```
void IsoTrackInfo::init(const RecTrack * const tr, int tptr, float tzvtx)
{
  init();

  chisq = tr->normalisedChiSquared();
  ptr = tptr;
  vhits = tr->foundHits();
  ihits = tr->lostHits();
  zvtx = tzvtx;

  TrajectoryStateOnSurface ts = tr->impactPointState() ;
  if (ts.isValid()&&
          !isnan(ts.globalMomentum().x()))
    {
      charge = tr->charge();
      mom.Value().setX(ts.globalMomentum().x());
      mom.Value().setY(ts.globalMomentum().y());
      mom.Value().setZ(ts.globalMomentum().z());
      tip = ts.globalPosition().perp();
      lip = ts.globalPosition().z();
    }
  else if((ts = tr->innermostState()).isValid() &&
          !isnan(ts.globalMomentum().x()))
    {
      cerr <<"IsoTrackInfo::warning, taking innermost state !!!" << endl;
      charge = ts.charge();
      mom.Value().setX(ts.globalMomentum().x());
      mom.Value().setY(ts.globalMomentum().y());
      mom.Value().setZ(ts.globalMomentum().z());
    }
  else if((ts = tr->outermostState()).isValid() &&
          !isnan(ts.globalMomentum().x()))
    {
      cerr <<"IsoTrackInfo::warning, taking outermost state !!!" << endl;
      charge = ts.charge();
      mom.Value().setX(ts.globalMomentum().x());
      mom.Value().setY(ts.globalMomentum().y());
      mom.Value().setZ(ts.globalMomentum().z());
    }
  else
    {
      cout << "IsoTrackInfo::warning, no valid state for track" << endl;
    }
}
```